SKROLLI 2018.1

SKROLLI

2018.1
SKROLLI.FI

Tietokonekulttuurin erikoislehti

Ohjelmoija, debuggaa itsesi

Näin lapsia opetetaan koodaamaan

Tampio ohjelmoi suomeksi

Pokitto Python Suomi-konsolissa

Rakenna uusi Commodore 64

QBASIC NERO 2000 FRACTALUS SKROLLI 5 V.

3 **Pääkirjoitus**

4 **Debuggaa itsesi**
Koodatessa voi oppia myös paremmaksi ihmiseksi.

9 **Kolumni: Janne Sirén**
Atari 2600:n E.T. olikin yllättäen aikansa paras lisenssipeli.

10 **Todellisuus lisääntyy**
Tärkeimmät lisätyn todellisuuden alustat.

12 **Writemonkey**
Askeettisen näköinen editori antaa kirjoittajan keskittyä oleelliseen.

14 **Vim**
Tekstieditoreista ei voi kirjoittaa ilman, että myös Vi(m) mainitaan.

15 **Python luikertelee Pokittoon**
Korkean tason kieli helpottaa lasten koodikokeiluita.

20 **Lapset ohjelmoijan opissa**
Epätoivoa ja onnistumisen iloa ohjelmointikerhossa.

25 **Kolumni: Jarno Niklas Alanko**
Bitcoinin kurssinousua kelpaa kirota katsomosta käsin.

26 **Uusnelonen**
Tiesitkö, että Commodore 64:ää valmistetaan taas?

32 **Sarjakuva**
Turrikaanien yössä: Brittien olematon sarjakuvaperinne pelitaiteena.

33 **Memotech MTX500**
Alumiinisten tietokoneiden pioneerin nimi alkoi kirjaimilla Mem, ei Mac.

34 **Fractalus**
Proseduraalinen maisemanluonti peli-ideana.

40 **Battletech**
Mech-pelihistorian lyhyt oppimäärä.

42 **Ei näin!**
Toimittajan matka e-urheilun maailmaan sai omituisia käänteitä.

46 **Nero 2000**
Digiarkeologit sukelsivat legendaarisen tietovisan saloihin.

52 **JOKstoriaa IV**
Se ensimmäinen oma kotimikro.

56 **Game Jam -viikonloppu**
Ideatehdas sykkii pelin 48 tunnissa.

60 **Olin teini-ikäinen QBasic-ohjelmoija**
BASIC ei sulattanutkaan aivoja.

64 **The Machine in the Ghost**
Robin Boastin kirja digitaalisuuden historiasta ja luonteesta.

65 **Disobey**
Selviytymisopas vuoden hakkeritapahtumaan.

68 **Neliöjuuritehtävän jälkipyykki**
Pähkinänurkassa tutustutaan klassikko-ongelman tuoreisiin ratkaisuihin.

69 **Ohjelmoi suomeksi**
Tampio-kielessä suomen sijapäätteillä on keskeinen rooli.

73 **Skrolli 5 vuotta**
Lehti tekijöiltä tekijöille. Mikä nimeksi?

Huomio! Skrolli-yhteisö jalkautuu tosielämään: Mika Hyvönen alias Mikron/Byterapers on Skrollin uusi yhteisömestari yli 20 vuoden tapahtumaorganisoinnin kokemuksella. Aikaisemman Skrolli Partyn ohelle on kehitteillä Skrolli-miitti eli Limiitti ja muuta. Seuraa siis Skrollin someja (skrolli.fi/yhteystiedot). Kiinnostaako yhteisötekeminen tai onko ideoita? Napauta postia: mika@skrolli.fi.

Fractalus, s. 34

Mikko Heinonen
vastaava päätoimittaja

Pois kaninkolosta

Selailin iltana muutamana kiinalaisen verkkokaupan Android-sovellusta, jossa eteeni vyöryi toinen toistaan erikoisempia joskin täysin tarpeettomia tavaroita. Kun napautin muutaman niistä auki lukeakseni lisätietoja, huomasin nopeasti etusivun luettelon alkavan muuttua. Minut oli pikavauhtia profiloitu ihmiseksi, joka haluaa katsella vain älykelloja ja nahkajäljitelmästä valmistettuja minihameita. Tämän jälkeen sain nähtäväkseni sivukaupalla samankaltaisia tuotteita. Asia alkoi hyvin pian harmittaa ja virtuaalinen shoppailumatka päättyi ilman kauppatapahtumaa.

Vaikka moni muu sovellus toimii hienovaraisemmin kuin itämaan kelpo kauppiaat, sama ilmiö on valloillaan ympäri internetiä. Kaikki merkittävät sosiaaliset mediat harrastavat samanlaista profilointia ja näyttävät asioita, joita ajattelevat ihmisten haluavan lukea tai katsoa. Omassa digitaalisessa kaninkolossa on turvallista ja tyytyväinen käyttäjä viipyy sivustolla pitempään sekä vierailee useammin. Samalla kuitenkin vähän kerrallaan eriydymme toisistamme omaan somaan algoritmikuoppaamme.

Skrolli ilmoittaa alaotsikossaan olevansa tietokonekulttuurin erikoislehti. Fraasi on alusta asti ollut kumarrus toisen harrastelehden, *Mobilistin* suuntaan. Sen päätoimittaja **Jan Enqvist** totesi minulle kerran, että lehden eräs tehtävä on kertoa ihmisille, mitä he haluavat lukea. Sisältöä ei voi päättää huutoäänestyksellä, sillä tällöin päädytään keskinkertaisuuteen ja kaikkien miellyttämiseen. Varastin ajatuksen välittömästi ja aloin toteuttaa sitä kuin omaani – myös Skrollia tehtäessä.

Olemme sikäli onnellisessa asemassa, että toimitukselle tarjotaan jatkuvasti erilaisia jutunaiheita ja tarinan raakileita. Niistä rakennamme teille nyt jo kuudetta vuosikertaa tätä aikoinaan nettivitsistä syntynyttä lukupakettia. Meidän tärkein tehtävämme onkin valikoida jokaiseen numeroon tarpeeksi erilaista asiaa, jotta mahdollisimman moni lukijoistamme kohtaisi sivuilla jotakin *yllättävää*. Tietotekniikan kentällä on liikaa kiinnostavaa asiaa, jotta suostuisimme lokeroimaan itsemme pelkäksi retrolehdeksi, pelilehdeksi, koodauslehdeksi, hakkerilehdeksi tai skenelehdeksi – vaikka samalla olemme aina vähän näitä kaikkia, jotta kaikki tuntevat olonsa myös kotoisaksi. Mutta vaikka kaninkolossa on mukavaa ja lämmintä, kelpo ristiturvan kannattaa välillä työntää nenäänsä myös naapurin puolelle. Tiedä, vaikka sitä innostuisi jostakin uudesta.

Skrolli
Tietokonekulttuurin erikoislehti

Yhteydenotot: toimitus@skrolli.fi
Ircnet: #skrolli
skrolli.fi

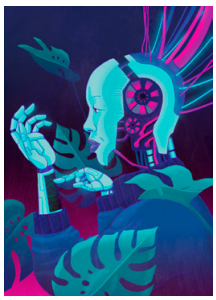
Vastaava päätoimittaja: Mikko Heinonen
Päätoimittaja: Tapio Berschewsky
Toimituspäällikkö: Valhe Kouneli
Taiteellinen johtaja, taitto: Nasu Viljanmaa
Digipäälliköt: Toni Kuokkanen, Janne Sirén
Taitto: Manu Pärssinen
Kuvatoimittaja: Laura Pesola
Mediamyynti: Jari Jaanto
Talous: Anssi Kolehmainen
Yhteisömestari: Mika Hyvönen

Muu toimitus: Jarno Niklas Alanko, Ville-Matias Heikkilä, Jukka O. Kauppinen, Ronja Koistinen, Sakari Lönn, Suvi Sivulainen

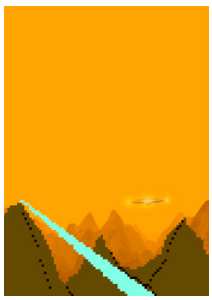
Tämän numeron avustajat: Heidi Akselin, Maxine Azzara, Ilkka Hauhio, Mikael Heikkanen, Ada-Maaria Hyvärinen, Otso Kassinen, Toni Kortelahti, Kari Lavikka, Sakari Leppä, Peter Levine, Tiina Malin, Aleksandr Manzos, Mitol Meerna, Tuukka Norri, Jarkko Nääs, Oona Räisänen, Jari Sihvola, Aki Sivula, Ismo Utriainen, Hannu Viitala

Julkaisija: Skrolli ry

Painopaikka: Hämeen Kirjapaino oy, Tampere,
ISSN 2323-8992 (painettu)
ISSN 2323-900X (verkkojulkaisu)



Etukannen kuva:
Mikael Heikkanen



Sisäkannen kuva:
Janne Sirén,
Fractalus-pelit

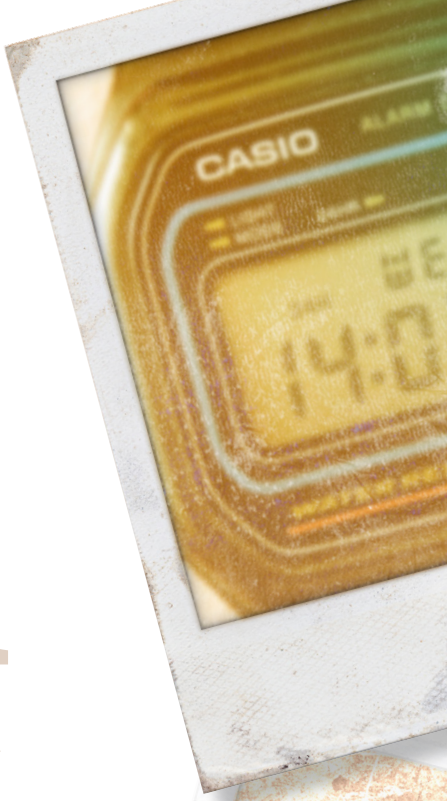
YMPÄRISTÖMERKKI
MILJÖMÄRKT
4041 0209
Painotuote
HÄMEEN KIRJAPAINO OY

DebuggaA itsesi

– erään ohjelmoijan tarina

Interview-lehden kannessa luki punaisella huulipunalla ”software”. Tiesin täsmälleen, mihin kaksimielisyyksiä harrastanut lehti sillä pyrki, mutta silti se onnistui lisäämään kiinnostustani tähän kiehtovaan, suorastaan mystiseen käsitteeseen.

Teksti: Pete Levine
Käännös: Mikko Heinonen
Kuvat: Wikimedia Commons (Konstantin Lanzet, Carlos Pérez Ruiz, Digon3, J.Stiegler, Liftarn, Keagiles), Manu Pärssinen



Tietotekniikassa oli kaksi keskeistä termiä: software, pehmo, joka tässä esitettiin feminiinisenä, ja sen vastine hardware eli kovo. Laitteisto oli helpompi käsittää, mutta mistä ohjelmoinnissa oikeastaan oli kyse? En osannut visualisoida tai selittää sitä, mutta tämä muodoton ”jokin” osasi simuloida ping-pongia, käyttää taskulaskinta tai vaikkapa muuttaa Casio-rannekelloni räiskintäpeliksi.

Halusin tietää asiasta enemmän, joten hakeuduin collegessa ohjelmointikurssille. Ohjelmiston näkymättömyys oli tehnyt siitä silmissäni maagista, ja odotin, että kurssilla minut vihittäisiin muinaisiin salaisuuksiin. Sain vastaani kuitenkin normaalia toimistotyötä. En odottanut, että kirjoittaisin käskyjä reikäkorteille ja odottaisin sitten kaksi päivää tuloksia. Jos tein yhdenkin virheen, koko prosessi piti aloittaa alusta. Tämä saisi riittää minulle, ajattelin – tai niin luulin.

Collegen jälkeen muutin Kaliforniaan täynnä unelmia ja suuria odotuksia, mutta olin jo hukannut aikaa kaikenlaiseen, ja se oli jättänyt jälkensä. Eläessäni käytännössä kädestä suuhun ja tehdessäni töitä vapaaehtoisena radiojuontajana tein sen johtopäätöksen, että ”pehmeiden” taitojeni sijaan tarvitsen jotakin ”kovempaa”, kuten radioinsinöörin osaamista. Radiolähetysten tekniikasta kiinnostuneena olin käynyt elektroniikan peruskurssin, joka tavallaan valmisti minut seuraavaan vaiheeseen. Ilmoittauduin ammattikouluun ja opin vahvistimista, modulaatiosta ja lähetystavoista. Opintojen loppuvaiheessa tutustuin Intel 8080 -mikroprosessoriin, joka oli ehditty tällä välin keksiä.

Jäin hetkessä koukkuun. Luin *8080 Programmers Reference Manualin* ja tunsin olevani velhon oppipoika. Olin lapsena lukenut taikatemppukirjoja, ja vaikka arvostin illuusioiden luomiseen vaadittavaa nokkeluutta ja temppuja, tieto myös lisäsi tuskaa. En koskaan voisi rakentaa laitetta, jolla saisin veljeni levitoimaan, enkä ollut tarpeeksi näppärä silmänkääntötemppuihin. Mikroprosessorin voima taas piili sen käskysetissä, sanaston muodostavassa matalan tason ohjelmointikielessä. Olin varma, että voisin oppia sen.

Luin manuaalin kannesta kanteen, en niinkään oppiakseni sitä ulkoa kuin tutustuakseni dokumentaation rakenteeseen, jotta tietäisin, mistä vastaukset löytyvät. Yhdistettynä väsymättömään haluuni etsiä tietoa tämä osoittautui hyväksi strategiaksi. Tein toki edelleen virheitä, mutta mikroprosessorin nopeuden ansiosta sain palautetta lähes välittömästi. Tämä nope-

ampi vasteaika antoi oppimiselleni potkua.

Nykyisin vaikkapa Googlelle voi päästä töihin ilman tietojenkäsittelytieteen tutkintoa. Jos osaat työn, saat sen. Menneisyydessä, ennen julkista internetiä, palkkaamiskäytännöt olivat huomattavasti jäykempiä, ja ammattikoulusta valmistuneelle vaihtoehtoja oli vähän. Voisin ryhtyä kenttäasentajaksi ja vaihdella laitteiden piirilevyjä, tai jos en haluaisi matkustaa, voisin mennä kokoonpanolinjalle ja tuijottaa suurennuslasin läpi päivästä toiseen. Yksikään työ, johon pääsin haastatteluun, ei sisältänyt ohjelmointia. Tarjolla olevat vaihtoehdot tuntuivat lohduttomilta, kunnes sain päähäni ottaa itse yhteyttä kansainvälisen puhelinyhtiön USA:n länsirannikon käyttökeskukseen. Yllätyksekseni sain sieltä työtä johtavan insinöörin assistenttina.

Kent Wilder oli vanhaa koulukuntaa. Hän rakensi tarvitsemansa työkalut, koska niitä ei vielä ollut olemassa ja koska *juuri tämä* oli hänen nähdäkseen insinöörin tehtävä. Ohjelmistosta ei näkynyt jälkeäkään, ja TOC:ssä (tekninen käyttökeskus) sijainneet kotitekoiset konsolit oli rakennettu ilman integroituja piirejä – vastusten, kondensaattorien ja induktorien pyhällä kolmiyhteydellä. Tämä tarkoitti tietyllä tavalla paluuta menneisyyteen, perusasioiden äärelle, mutta en välittänyt siitä. Kent oli juuri sellainen mentori, jonka tarvitsin. Hän oli kärsivällinen ja pitkämielinen ongelmanratkaisija, joka lähestyi haasteitaan tyynesti, olivatpa kyseessä sitten kytkentävirheet Malesiassa tai New Yorkin insinöörien lähettämät pisteliäät kommentit.

TOC oli jaettu kolmeen alueeseen: konsolihuoneeseen, josta kytkentöjä valvottiin, tasaimia ja multipleksereitä täynnä olleeseen laitehuoneeseen sekä johdotushuoneeseen, joka yhdisti eri osat. Uuden piirin lisääminen tarkoitti, että johto piti juottaa tiettyyn kiinnitysnapaan, vetää tiukalle ja pujottaa sitten haluttuun kohteeseen. Kokonaisessa piirissä tällaisia kytkentöjä piti tehdä vähintään tusinan verran. Vaikka johdotushuoneen jakokehikot pyrittiin pitämään siistinä, ne olivat kolme metriä korkeita ja tiheänään punavalkoista kierrettyä parijohtoa.

Kehikoiden määritykset tallennettiin kansistaan pullisteleviin irtolehtimappeihin. Teoriassa välipisteitä ristiin vertaamalla oli mahdollista selvittää tietyn piirin reitti keskuksen sisällä, mutta sen alkupistettä, kohdetta tai keskuksen seurantapisteitä ei voinut tunnistaa miten-

kään. Ne olivat olemassa ainoastaan Kentin piirustuksissa.

En siis yllättynyt kuullessani, että taloon oli palkattu ohjelmoija: jos missään koskaan tarvittaisiin tallennusjärjestelmän uudistusta niin täällä. Kyselin ohjelmoijalta kunnioittavasti hänen projektiinsa liittyviä kysymyksiä, mutta kohteliaan ulkokuoreni alla sykki itseoppineen ohjelmoijan ylimielinen sydän, sillä olin hiljattain hankkinut Sinclair ZX-81:n ja opetellut BASICia.

Kun sain tietää, että projekti oli vain sähköinen muistilehtiö eikä kattava ratkaisu infrastruktuurimme luettelointiin, suuruudenhulluuteni pääsi valloilleen. Otin pyhäksi tehtäväkseni pelastaa yhtiön kaaokselta. Jos *hän* ei aikonut – tai osannut – tehdä sitä, niin *minä* tekisin, koska *minä osasin*. Ilmoittauduin anonyymisti vapaaehtoiseksi tehtävään. En kertonut kenellekään mitä tein, en edes Kentille, mutta otin tehtäväkseni analysoida, luokitella ja dokumentoida San Franciscon TOC:n laitteet ja piirit.

Työstin raporttia kotonani joka ilta useiden viikkojen ajan. Käytössä ei ollut tekstinkäsittelylaitetta, joten kirjoitusvirheiden sattuessa kirjoitin koko sivun uusiksi. En osannut arvioida, kauanko toteuttaminen kestäisi tai paljonko se maksaisi. En välittänyt siitä, koska halusin tehdä sen ilmaiseksi ja käyttää niin paljon aikaa kuin oli tarpeen. Tästä oli tullut kohtaloni.

Kun sain ehdotukseni valmiiksi, siinä oli yli 100 sivua. Sidoin raportin harmaisiin kiiltopahvikansiin ja esittelin sen TOC:n päällikölle. En odottanut saavani vastausta heti, mutta kun oli kulunut jo useita päiviä eikä mitään kuulunut, kokosin rohkeuteni ja kävin koputtamassa toimiston ovelle. Päällikkö poimi kansion pöydältä ja palautti sen minulle kuin lainaamansa kynän. ”Kivat kannet”, hän totesi, ”mistä ostit ne?” Ja siinä se. Ei sanaakaan ideoistani tai ehdottamistani ratkaisuista. Oliko hän edes lukenut sitä? Olin sanaton. Lähdin toimistosta loukkaantuneena.

Vaikka olinkin tullut täydellisesti torjutuksi, olin kehittänyt elegantin ratkaisun, hionut teknisen kirjoittajan taitojani ja saanut tyydytyksen suuren projektin loppuun saattamisesta. Olisin kuitenkin toivonut myös hieman tunnustusta. Sitä paitsi tosiasiassa en koskaan halunnut tekniseksi kirjoittajaksi, ja mitä hienoa ilmaiseksi työskentelyssä on?

Opinko siis tästä kokemuksesta mitään – sen tärkeän muistutuksen lisäksi, että TOC:n päällikkö oli ääliö?

Kyllä opin.

(Opetus 1 - Sisäänhengitys)

Opin, että hyväksyntä on psyykelleni niin tärkeää, että valjastan perfektionismini epätoivoisiin projekteihin, joista en saa juuri lainkaan hyötyä itselleni. Teen näin, jotta saisin elämälleni tilapäisen tarkoituksen ja välttyisin tajuamasta sitä, että eristäytymällä tällä tavoin omaehtoisesti vain pahennan jo ennestään yksinäistä elämää, jonka perusta ei ole kestävä.

(Opetus 1 - Uloshengitys)

Opin tämän, mutta en heti, sillä olin mieltynyt ZX-81:n ohjelmointiin ja näin mahdollisuuden käyttää sitä *toiseen* töissä kohtaamaani ongelmaan.

Minun piti vain suunnitella ZX-81:lle lisäkortti, asentaa komponentit kortille, tehdä johdotukset ja kirjoittaa ohjelma, joka lukisi, taulukoisi ja muotoilisi tiedot ja tulostaisi raportin. Siinä kaikki. Niin, ja sitten piti estää ZX-81:n ylikuumeneminen ja lisäkortin irtoaminen sekä näytön (kanavalle 3 viritetty matka-tv) sekoaminen. Työstin tätä kuukausikaupalla kärsivällisesti, miltei mielipuolisesti, ja tallensin ohjelman jokaisen version kasetille, kunnes lopulta tunnustin rakkaan pikku koneeni rajoitukset ja hyväksyin sen, ettei tämä projekti onnistu.

Opinko siis tästä kokemuksesta mitään, sen lisäksi, että tajusin miten typerää on käyttää näin paljon aikaa ja vaivaa johonkin, mitä kukaan ei edes pyytänyt? Kyllä opin.

(Opetus 2 - Sisäänhengitys)

Opin, että huono itsetunto synnyttää defensiivisyyttä – olisin tehnyt mitä tahansa ansaitakseni Kentin kunnioituksen – ja kilpailuhenkisyyttä, sillä halusin ehkä myös syrjäyttää hänet tuomalla TOC:hen digitaalitekniikkaa. Valitsemalla mahdottomia tehtäviä vahvistin riittämättömyyden tunnettani samalla kun kannoin kaunaa siitä, että panostuksiani ei tunnustettu.

(Opetus 2 - Uloshengitys)

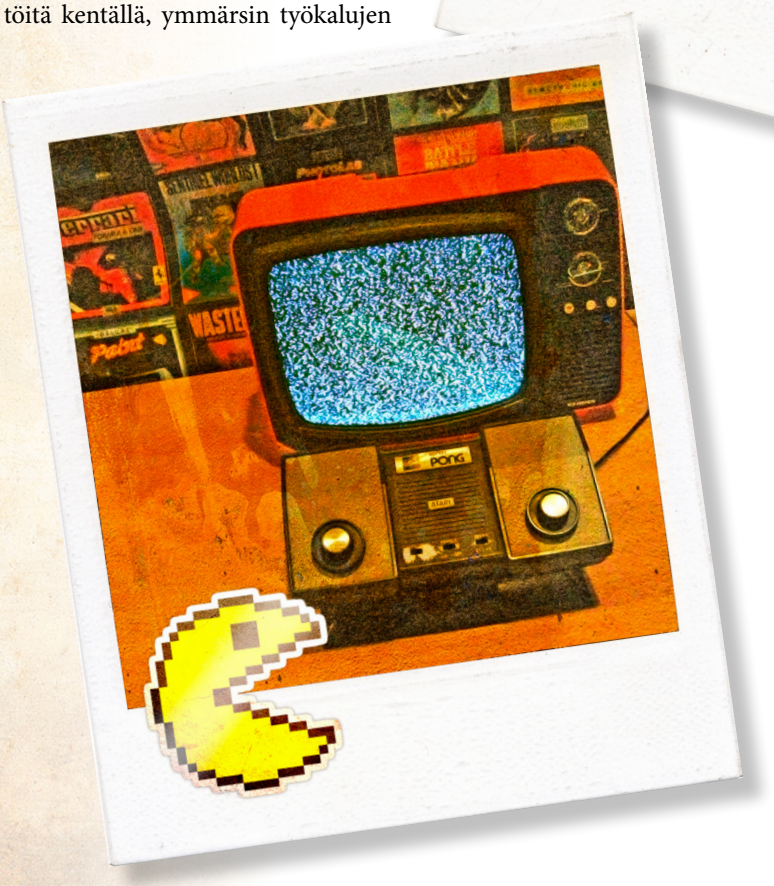
Opin tämän, mutta en heti, sillä oli aika siirtyä eteenpäin. Ohjelmiston läheisyyden, työsuhdeuima-altaan ja gourmet-kanttiinin houkuttelemana lähdin Piilaaksoon töihin yhtiöön, joka suunnitteli ja valmisti tietokonepohjaisia puhelinkeskuksia. Haittapuolena oli yli kahden tunnin työmatka ruuhkassa. Pääsin sisään puhelinyhtiökokemukseni ansiosta.

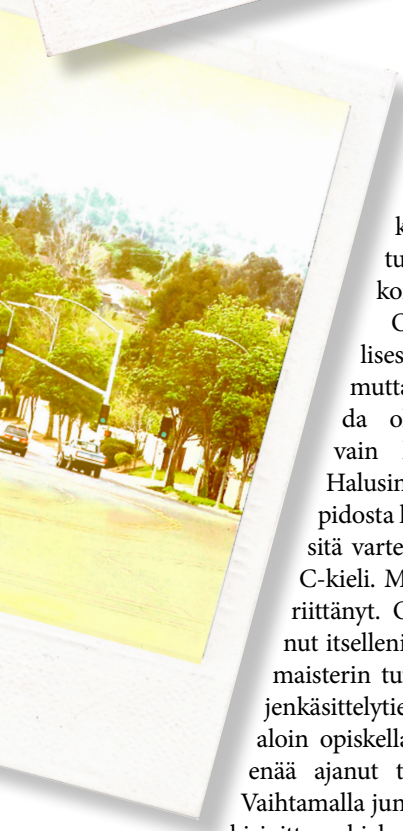
Aloitin työt käyttöinsinöörinä, joka vastasi kenttäasentajien puheluihin ja jalkautui ajoittain heidän luokseen paikan päälle. Jos ongelma vastasi tunnettujen bugien listassa olevaa kuvausta, asensin uudempaa versiota odotellessa tilapäisen ”patchin”, joka ohitti virheellisen kohdan. Tällainen patchaaminen oli riskaabelia, sillä minun piti kirjoittaa konekieltä käynnissä olevaan järjestelmään epäluotettavan modeemiyhteyden kautta. Lyöntivirheet saattoivat aiheuttaa katastrofin.

Ryhmämme kokenein patchaaja oli **Mark**, joka tunnettiin loukkaavasta käytöksestään. Hänen suosikkiheittonsa oli lievästi homofobinen ”käypä polvillesi”. Hän oli kiusaaja, joka panttasi tietojaan ja pilkkasi kaikkia, jotka eivät tienneet yhtä paljon. Asioiden kysyminen häneltä oli alentavaa, mutta siedin sen. Lopulta opin itsekin kirjoittamaan patcheja ja heti sen jälkeen pyysin siirtoa pois ryhmästä.

Uusi työni oli ylläpidossa, jossa korjasin vanhemman mutta vielä käytössä olevan ohjelmiston bugeja. Tähän ei ole olemassa koulutusta, pitää vain tutkia koodia ja lukea kriittisellä asenteella dokumentaatiota ja kommentteja. Koodin kirjoittanut ohjelmoija ei ole enää töissä yhtiössä tai hän on liian kiireinen – tai sitten hän on unohtanut, mitä teki. Olet omillasi, ja aika pian sinusta itsestäsi tulee auktoriteetti asiassa.

Jotkut minulle määrätyistä bugeista olin välittänyt eteenpäin kentältä, mutta muutamat olin myös itse löytänyt diagnostiikkaa tehdessäni. Koska olin itse tehnyt töitä kentällä, ymmärsin työkalujen





tärkeyden ja koin henkilökohtaista vastuuta niiden korjaamisesta.

Olin nyt virallisesti ohjelmoija, mutta halusin luoda ohjelmia enkä vain korjata niitä. Halusin siirtyä ylläpidosta kehitykseen, ja sitä varten piti opetella C-kieli. Mutta sekään ei riittänyt. Olin vakuuttanut itselleni, että tarvitsin maisterin tutkinnon tietojenkäsittelytieteestä, joten aloin opiskella kursseja. En enää ajanut töihin autolla. Vaihtamalla junaan sain aikaa kirjoittaa ohjelmia, joita sitten kokeilin töiden jälkeen toimistossa tai koulun mikroluokassa, koska minulla ei vielä ollut PC:tä kotona. Useammin kuin kerran myöhästyin viimeisestä junasta ja nukuin toimistolla. Aikataulu oli rankka ja vaivannäkö täysin tarpeetonta, koska kukaan ei vaatinut minulta maisterin tutkintoa. Lopulta päätin, että voisin edetä ilman sitäkin, ja tein niin. En ollut enää pelkkä ohjelmoija: sain siirron kehityspuolelle, ja minusta tuli ohjelmistoinsinööri, suunnittelija, luoja.

Osastolla oli pakkomielle kokouksista ja metodologiasta. Suunnitteluasiakirjoja arvioitiin muodollisissa tilaisuuksissa paitsi logiikkavirheiden, myös kieliopin ja muotoilun kannalta. Vikojen etsiminen oli nyt elämäntapa, saimme palkkaa siitä, ja kriittiselle luonteelleni tämä sopi erinomaisesti. Meiltä odotettiin täydellisyyttä, joten kohteliaisuuksia ei juuri jaeltu. Opimme kuitenkin ottamaan vastaan kovaakin kritiikkiä loukkaantumatta siitä, ja matkan varrella myös ryhmähenki kehittyi.

Tätä yhteistyön henkeä vahvisti UNIX-kehitysympäristö, joka rohkaisi jakamiseen ja avoimuuteen korostamalla yksinkertaisuutta ja modulaarisuutta. Se oli jotain aivan muuta kuin Markin tietojen panttaus tai Microsoftin läpinäkymättömyys. Salaisuuksia ei ollut. UNIXin man-sivujen kattava dokumentaatio oli kaikkien katsottavana. Jos et ollut valmistautunut huolellisesti, kehotus ”RTFM” tuli tutuksi.

Viehätykseni Pongiin oli muuttunut mielenkiinnoksi käyttöliittymiä kohtaan, ja otin tehtäväkseni ”frontendin” kehittämisen monimutkaiselle puhelinkeskussovellukselle. UI-konseptityöpajan suorittamisen jälkeen olin vakuuttunut, että PC-sovellusten pitäisi noudattaa Windowsin ilmettä. Koska sovelluksemme oli tarkoitus esittää ”tyhmillä” päätteillä, joissa ei ollut hiirtä, tämä oli aika väkinäinen tavoite. Ajattelin kuitenkin, että se tekee lopputuotteesta visuaalisesti uskottavan ja sementoi myös minun asemani UI-osaajana. Teknisestä näkökulmasta tämä osoittautui huomattavan työlääksi. Työpäivät venyivät, kun huomasin kerta toisensa jälkeen maalanneeni itseni nurkkaan ja jouduin pelastautumaan sieltä yhä monimutkaisempien keinojen avulla.

Suurempi haaste oli kuitenkin konseptitasolla: siinä, miten sovellus esitellään vähemmän tekniselle ihmiselle. Ilman älykästä frontendiä järjestelmän käyttäminen olisi vaatinut useiden kryptisten komentojen ja parametrien kirjoittamista, minkä lisäksi olisi pitänyt tuntea noin puolen tusinan muuttujan väliset suhteet. Markkinointi ei ollut ottanut tätä puolta asiasta lainkaan huomioon, mutta itse näin sen ydintehtävänäni, koska ajattelin asiaa käyttäjän enkä ohjelmoijan näkökulmasta. Minun piti rakentaa näille erillisille elementeille yhtenäinen viitekehys ja piilottaa ne käyttöliittymän taakse. Taustalla koodiliima sitten muuttaisi sen oikeiksi ”backendin” komennoiksi.

Hyvinä päivinä koin olevani hyväntahtoinen diktaattori, jonka pienet alamaiset reagoivat virheettömästi näppäimistöltä kirjoittamiini käskyihin. Huonoina päivinä, jolloin en onnistunut korjaamaan itse tekemääni virhettä ja istuin yhä toimistossa kymmeneltä illalla – tietäen, että töissä piti olla jälleen seitsemältä aamulla – kyseenalaistin kaiken mitä tein.

Syytin itseäni myös virheistä, joita en ollu tehnyt. Erään poikkeuksellisen turhauttavan istunnon aikana tuntui, kuin itse logiikka olisi kääntynyt minua vastaan. Kävin koodia läpi rivi kerrallaan ja yritin ymmärtää, miksi muuttuja ei välillä päivittynyt oikein. Turhautuneena käynnistin tietokoneeni uudelleen, ja ongelma katosi. Syynä oli ollut tilapäinen sähköinen häiriö, ei vika koodissani.

Projekti jatkui kuukausia, ja tein säännöllisesti 10–12 tunnin päiviä. Avioliittoni kärsi, mutta olin jo pakkomielteinen asian suhteen ja vakuuttunut, että kun hanke valmistuu, saan haluamani ja ansaitsemani ylennyksen. Niin ei käynyt. Tuloksena olikin projektinjälkeinen masennus, joka muuttui pian oikeaksi masennukseksi.

Opinko tästä kokemuksesta mitään, lukuun ottamatta sitä että viiden tunnin yöunet eivät riitä, ja että minun olisi pitänyt kuunnella vaimoani ja olla ottamatta vastaan työtä, jossa työmatkat ovat näin pitkiä?

Kyllä opin.

(Opetus 3 - Sisäänhengitys)

Opin, että liioittelullani ei ole mitään luontaista ylärajaa, ja että kuvittelemalla itseni oletetun loppukäyttäjän asemaan – ja ihan vain olemalla ylimielinen – olin kadottanut käsityksen työkuormituksestani ja aliarvioinut projektin vaatiman ajan ja resurssit. Samaan aikaan olin eristäytynyt muista niin, että poliittisesti taitavammat ja perinteisemmin kunnianhimoiset kollegani olivat päässeet etenemään urallaan.

(Opetus 3 - Uloshengitys)

Opin tämän, mutta en heti, sillä oli jälleen aika edetä. Miksi juuttua antiikkisen teknologian pariin firmassa, jossa minua ei arvostettu, kun internet kasvoi vauhdilla ja ketterät uudet yri-

tykset keräsivät rahat? Vaihdoin siis oman toimiston karsinaan startup-yrityksen avokonttorissa kotini lähellä ja sain sen mukana optioita, oluenhuuruisia juhlia ja nopeaa kehitystä. Kaikki meni hyvin puolitoista vuotta, kunnes yritys menetti suurimman asiakkaansa, optioista tuli arvottomia ja yritys irtisanoi puolet henkilöstöstään – minut mukaan lukien. Tällä kertaa oppi meni perille välittömästi, kun muistin isäni sanat: *Jos jokin vaikuttaa liian hyvältä ollakseen totta, se todennäköisesti on sitä.*

Näppini polttaneena halusin enemmänkin turvallisen työpaikan, joten hakeuduin Fortune 10 -yhtiöön. Minut palkkasi UNIX-konsultiksi tietotekniikkaa osaamaton päällikkö, jonka kunnianhimoisten selkäänpuukottajien tiimille ajatuskin muutoksesta oli kauhistus. Tein analyysejä ja kirjoitin muutaman aikaa säästävän ohjelman, mutta myös sähläsin sekä ihmissuhteiden että teknologian tasolla. Työyhteisö oli paikalleen juuttunut ja pikkumainen, eivätkä he pitäneet uudesta tulokkaasta. Minulla taas ei ollut lainkaan Kentin tyyneyttä, vaan olin ylimielinen ja varomaton. Käytössä ei ollut erillistä testiympäristöä, ja valitettavasti kokeilin erästä ohjelmaani yöllä tuotantoympäristössä. Aivan satunnainen asia, jota en ollut ottanut huomioon, kaatoi koko yhdyskäytävän, ja sain potkut. Opinko nyt mitään? Voi kyllä.

(Opetus 4 - Sisäänhengitys)

Pikku lammikoissa uivat isot kalat eivät arvosta kalatieteilijää.
Kaiken kyseenalaistaminen ei ole keino saada ystäviä.
Vaikka oletkin ylimielinen, koska olet erinomainen työssäsi, olet silti ylimielinen.
Tuotantojärjestelmällä ei testata.

(Opetus 4 - Uloshengitys)

En ollut ainoa työtön, sillä dotcom-kupla tyhjeni edelleen. Hakemuksiin ei vastattu, ja haastatteluihin oli vaikea päästä. Mutta kun työmarkkinat lopulta avautuivat, olin valmiina. *Olin kuin olinkin oppinut jotain.* Jätin Piilaakson ja pitkät työmatkat. Jätin 12 tunnin päivät ja projektit, jotka valtasivat ajatukseni ja elämäni. Halusin stressittömän työn läheltä kotoa ja kahdeksan tunnin työpäivät. Minulle kävisi, että palkka ei olisi yhtä hyvä kuin aiemmin. Olin nimittäin tajunnut, että itse projektit eivät olleet vahingollisia, vaan minä itse vedin aina ikiliikkujan kammesta käyntiin. Siksi minun piti myös vetää jarruvipua. Löysin haluamani sanfranciscolaisesta sairausvakuutusyhtiöstä.

Haastattelussa tehtävä oli kuitenkin vaikuttanut teknisemmältä kuin se olikaan. Työ, jossa piti varmistaa, että vakuutettujen nimet ja taustatiedot oli lähetetty tietokantaan oikein, oli lopulta melko kuivakkaa. Analyysiksi kutsuttu työ olikin vain tietojen vertailua. Kyllästyin melko pian.

En pidä kyllästymisestä, ja ikävystyttävä työ alkoi veltostuttaa muutenkin. Siispä ryhdyin tuotteliaaksi ja aloin kirjoittaa ohjelmia, jotka tekisivät työn puolestani. Halusin niiden olevan helppokäyttöisiä ja vaativan mahdollisimman vähän käyttäjän toimia. Annoin niille näppärät nimilyhenteet, jotta päällikköni voisi kertoa niistä omalle päällikölleen, ja esittelin ne sitten omalle tiimilleni.

Ohjelmat puolittivat työkuormani. Tästä eteenpäin minun ja tiimini päätehtävä oli – Facebookin selailun lisäksi – varmistaa, ettei päällikkö saa tietää, kuinka paljon vapaata aikaa meille itse asiassa jäi. Käytin omaa aikaani novellien kirjoittamiseen ja tein myös lisää ohjelmia, joita arvostettiin ja joista sain bonuksia.

Lisää tyydytystä sain siitä, että järjestelmä ei enää seonnut, jos saman perheen jäsenten nimien alussa oli kolme samaa kirjainta. En tietenkään voinut vaikuttaa siihen, jos heiltä myöhemmin evättiin korvaukset, mutta ainakaan se ei johtunut kirjausvirheestä.

Olen nyt eläkkeellä, mutta pyrin yhä torjumaan oppimiani huonoja tapoja. Ohjelmiston virtuaalisessa maailmassa hallinta on keskeinen käsite. Olet kuin Aladdin ja ohjaat lampun henkeä, joka haluaa täyttää jokaisen toiveesi, tai kuin pikku Napoleon armeijoineen. Arvioitaessa ohjelmistoa olet saivartelija, jonka odotetaan puuttuvan pienimpäänkin yksityiskohtaan.

Elämä ei kuitenkaan ole ongelma, jota pitää hallita tai korjata, eikä tällaisella asenteella voi saada ystäviä tai vaikuttaa muiden mielipiteisiin. En ole enää työnarkomaani, mutta kuitenkin taipuvainen pakkomielteisiin. Yritän suunnata energiani avioliittooni ja kotiimme. En enää vaadi täydellisyyttä, vaikka toki pidän sitä ohjenuorana. Pyrin pysymään oikealla tiellä, mutta ajoittain joudun eksyksiin ja minut pitää bootata. Syvä hengitys riittää, jos olen oikeassa mielentilassa – muuten tarvitsen hellän läpsäytyksen. Paraskin ohjelmisto nimittäin oikuttelee välillä.

Soitto kotiin

Päinvastoin kuin väitetään, Atari 2600:n E.T. oli aikansa paras lisenssipeli.

Janne Sirén

”I’ll be right here”, sanoi *E.T. the Extra-Terrestrial* -elokuvan (1982) nimihahmo, kosketti vastanäyttelijän otsaa hehkuvalla sormellaan ja syöpyi sukupolvien muistiin. E.T. oli 1980-luvun suosituin elokuva, suositumpi kuin *Tähtien sodat*. E.T.:stä ei tehty jatko-osia, eikä etko-osia, mutta ehkä juuri siksi se on kestänyt aikaa – eikä se ole unohtunut.

Unohtunut ei ole myöskään elokuvasta tehty lisenssipeli, *E.T. the Extra-Terrestrial* Atari 2600 -pelikonsolille (1982). Useimmat pelaajat tuntevat surullisen tarinan: Rekkalasteittain myymättä jääneitä E.T.-pelejä haudattiin New Mexicon erämaahan jo 1983. ”I’ll be right here”, sanoi E.T.-pelimoduuli, kunnes kaivausryhmä kaivoi sen esiin vuonna 2014.

Kaikkihan sen tietävät: E.T. oli maailman huonoin videopeli ja tappoi varhaiset pelikonsolit.

Myytinmurtajaiset

Myytillä tarkoitetaan yleistä, mutta harhaista uskomusta. Esimerkiksi E.T.-elokuvasta muistetaan E.T.:n keksineen sanat ”E.T. phone home”. Oikeasti E.T. sanoi ”E.T. home phone”. Gertie – **Drew Barrymore** – sanoi ”E.T. phone home” (E.T. toisteli myöhemmin Gertien sanoja). Myös tarinaa E.T.-pelimoduuleista erämaassa on dramatisoitu. Kyseessä oli kaatopaikka Alamogordon kaupungissa.

Entä se peli: mikä on myyttiä ja mikä totta? Kiistatonta on, että E.T.-peli sai alkunsa kiireessä. Pelin Atarille toteuttanut **Howard Scott Warshaw** sai vain viitisen viikkoa aikaa sen toteuttamiseen, vaikka tyypillinen toteutusprojekti kesti tuolloin puoli vuotta. Lisenssineuvottelut olivat venyneet, ja peli piti saada joulumarkkinoille. Vertailun vuoksi pelimoduulien valmistus kesti kymmenen viikkoa.

Pelin toteutus alkoi pikasuunnittelulla. Warshaw hahmotteli vapaasti kuljettavan pelimaailman esteineen sekä tehtävän: E.T. saapuu Maahan, etsii osat puhelimeensa, soittaa kotiin ja alus noutaa hänet. Ihmishahmoja olisi pelissä elokuvaa mukailevissa sivurooleissa. Ohjaaja **Steven Spielberg** ei suunnitelmasta innostunut, vaan kannatti E.T.-henkistä Pac-Mania.

Warshaw kuitenkin uskoi oman konseptinsa olevan lähempänä elokuvaelämystä ja sai tahtonsa läpi. Kotiin soittava E.T. ehti kauppoihin jouluksi 1982. Atari oli yltiöoptimistinen ja valmisti peliä arviolta neljä miljoonaa kappaletta.

Epätodennäköinen esikuva

Muistan hämärästi ensikokemukseni E.T.:stä 80-luvulta. Reaktioni oli sama kuin monilla ajan lapsilla: hämmennys. E.T. ei ollutkaan suoraviivainen toimintapeli, siinä ei itse asiassa ole väkivaltaa lainkaan. Epälineaarinen pelimaailmakin oli uutta. Ohjeiden tulkkaaminen auttoi, mutta tämä jäi monilta tekemättä. Spielbergin Pac-Man olisi ollut lähempänä odotuksia.

Kiirehtiminen oli myös jättänyt E.T.:hen useita pieniä ongelmia. Näitä eräs pelin fani listasi ja myös korjasi ansiokkaasti Neocomputer.org-sivustolla vuonna 2013. Yksi tällainen on pelin kirjaimellinen törmäysrutiini, jonka vuoksi E.T. osuu ja putoaa kaivoon, vaikka vain sen pää olisi kaivon yläpuolella. Faniversiossa tämä on korjattu.

80-luku muistetaan löyhästi elokuviin perustuvista tusinapeleistä, jotka kutistivat tarinat irrallisiksi toimintapelikohtauksiksi. E.T.:n alustatoveri, samana vuonna ilmestynyt *Star Wars - The Empire Strikes Back* (ks. Skrolli 2013.3), oli malliesimerkki: kokoillan elokuvasta peliin selvisi vain päättymätön lumikiituriammuskelu.

E.T. oli kunnianhimoisempi. Se pyrki pelillistämään saumattoman lisenssitarinan alusta loppuun, mikä oli uraauurtavaa. Lopussa kiitos seisoo – nykypäivän elokuvapelit ovat tässä lähempänä E.T.:tä kuin monia muita kasaripelejä. Eikä E.T. huonosti pärjännyt aikanaankaan: 1,5 miljoonaa myytyä kappaletta oli lähellä Atari 2600:n parhaimmistoa ja puoli miljoonaa enemmän kuin Empire Strikes Backilla.

Peliä toki valmistettiin ihan liikaa, mutta lopulta E.T.:n suurin synti oli syntyä väärään aikaan. Se ilmestyi Commodore 64:n julkaisuvuoden lopulla. New Mexicon hiekkaan päätyi pian paljon muutakin. Kotimikrovallankumous oli alkanut.

Linkit E.T.-peliin Internet Archivessa sekä muunnelmaprojekteihin: skrolli.fi/numerot

Todellisuus lisääntyy

Darth Vader kävi meillä

Maailman ensimmäinen virtuaalikypärä oli samalla lisätyn todellisuuden silmikko. Nyt lisätystä todellisuudesta puhutaan taas. Skrolli listasi tärkeimmät AR-alustat.

Teksti: Janne Sirén
Kuvat: Janne Sirén, Microsoft, IKEA, HoloKit, Lenovo, Vuzix

Puristan kaksin käsin metallisen väristä muoviputkiloa. Katson ympärilleni: Oma koti. Sohva ja pöytä. Pino Skrolleja ja eilisen tiskit. Sohvan vieressä seisoo Darth Vader. Puristan kovempaa, ja muoviputkilosta nousee sinertävä valopatsas. Darth Vader nostaa oman, punaisen valomiekkansa…

Kun kerroimme virtuaalitodellisuuden 150-vuotisesta historiasta Skrollissa 2016.2, kerroimme myös *The Sword of Damocles* -nimisestä virtuaalikypärästä vuodelta 1968. Damokleen miekkaa pidetään ensimmäisenä tietoteknisenä keinotodellisuussilmikkona. Silmikko toimi myös läpinäkyvänä, jolloin vektorigrafiikka näyttäytyi tosimaailman keskellä. Tätä kutsutaan lisätyksi todellisuudeksi.

Keinotodellisuudessa näyttää esiintyvän kypäriä ja miekkoja. Hienostuneita aseita sivistyneemmälle ajalle, vaiko edelleen pelkkä kangastus?

Microsoftin sekatodellisuus

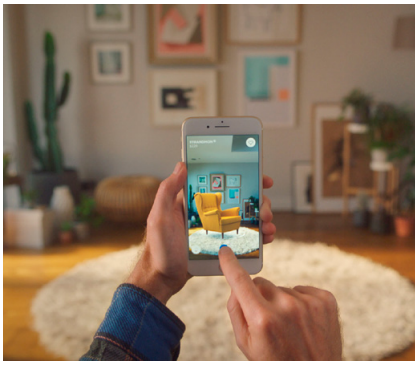
Virtuaali- ja lisätty todellisuus saivat alkunsa samanaikaisesti. Virtuaalitodellisuudella (*virtual reality, VR*) tarkoitetaan tosimaailman peittävää lumetodellisuutta. Lisätty todellisuus (*augmented reality, AR*) taas on keinotodellisuuselementtien näyttäytymistä todellisuudessa. Yhteistä on liikkeenseuranta: tekotodellisuus näyttää aidolta vain, kun se liikkuu uskottavasti katseen ja tilan mukana.

HoloLens (3 349 euroa).

Tähän saakka kuluttajatuotteiden kehitys on kuitenkin ollut erillistä: lisättyä todellisuutta on nähty lähinnä kannettavien peli- ja älylaitteiden kamerasovelluksissa, virtuaalitodellisuutta pelisilmikoissa. Kerroimme esimerkiksi Googlen erillisistä Daydream (VR)- ja Project Tango (AR) -hankkeista Skrollissa 2017.1.

Ensimmäisenä näitä erillistodellisuuksia kiirehti yhdistämään Microsoft. Windows 10:n versio 1709 (Fall Creators Update, 2017) sisältää Windows Mixed Realityn eli ”Windowsin yhdistetyn todellisuuden”, joka kattaa molemmat keinotodellisuuden muodot. Microsoftin kumppanit julkaisivat samalla joukon Mixed Reality -silmikoita, joiden liikkeenseuranta perustuu kameroihin eikä vaadi ulkoisia majakoita aiempien PC-lasien tapaan.

Microsoftin sekatodellisuus on kuitenkin toistaiseksi vain lupaus tule-



ARKit ja IKEA Place. Lume-IKEA, jossa huonekaluja ei tarvitse koota itse.

Mirage AR (199 euroa).
Myös Kylo Ren esiintyy Jedi Challengesissa.

HoloKit (35 dollaria) on AR-pahvisilmikko iOS:lle ja Androidille.

vasta. Silmikoista ainoastaan kallis ja ikääntyvä ammattilaistuote HoloLens tukee läpinäkyvää lisättyä todellisuutta. Siinä on myös pelkkä klikkeriohjain, eikä tukea monipuolisille liikeohjaimille. Mixed Reality -silmikot taas ovat vasta ”pelkkiä” virtuaalilaseja.

Esimakua lisätystä Windows-todellisuudesta tarjoaa sentään Microsoftin Mixed Reality Viewer -sovellus, joka piirtää 3D-kuvia tietokoneen kamerakuvan keskelle.

Applen ja Googlen tanssi

Google puolestaan on esitellyt Tango-projektiaan vuosien ajan. Projekti tutki älylaitteen sijainnin havainnointia tilassa ja tuotti vaikuttavia AR-demoja. Tangon kyvyt perustuivat älylaitteeseen integroituun kamerapatteristoon, joka näkee tarkalleen, missä ja missä asennossa laite on, ilman ulkoisia majakoita. Pari yhteensopivaa laitettakin julkaistiin.

Tosissaan lisätyn todellisuuden kilpa kuitenkin käynnistyi, kun Apple paljasti ensimmäiset keinotodellisuuskorttinsa julkaisemalla ARKit-kehitysalustan iOS-laitteille kesäkuussa 2017. Toisin kuin Tango, ARKit ei vaadi erityiskameroita, vaan älylaitteen perussensorit riittävät AR-sovellusten toteuttamiseen. Uudemmat iPhonet ja muutamat iPadit kelpaavat laitteiksi.

Googlen vastaus tuli elokuussa: ARCore-alusta tavallisille Android-älypuhelimille, joka korvaa Tangon. Eräs kehittäjä kuvaili ARCorea sanoin ”uudelleennimetty Tango, josta on kommentoitu pois erikoiskameravaatimus”. Tangoa varten kehitetty kamerakonsepti ei silti ole häviämässä – se on osa Googlen tulevan Daydream Standalone -VR-alustan WorldSense-liikkeenseurantaa.

ARKit ja ARCore/Tango toimivat uudehkojen älylaitteiden ruudulla, kamerakuvan päällä. Virallisia AR-laseja ei niillekään saa. Pahvinen HoloKit-silmikko tosin tarjoaa ensiapua.

Lenovon kangastus

Windows Mixed Realityn keskittyessä virtuaalisilmikoihin ja mobiilijättien lisätessä todellisuutta vasta älylaitteen ruudulle, musta hevonen tulee Lenovolta. Lenovon Mirage AR (aloituskuvassa) on lisätyn todellisuuden silmikko älypuhelimille. Kuten HoloKitissä, puhelin kiinnitetään heijastusnäytöksi silmikon sisään.

Erityiseksi Mirage AR:n tekee sen ensisovellus: *Star Wars – Jedi Challenges* -peli valosapeleineen. Peli on pakattu silmikon ja liikkeenseurantamajakan mukaan. Peli sisältää valomiekkataistelun, strategiapelimäisen maataistelun ja holoshakin. Ja kun sanomme ensisovellus, tarkoitamme ainoa sovellus. Mirage AR ei ole yhteensopiva minkään AR-alustan kanssa.

Jedi Challenges toimii jotenkuten. Darth Vaderia ja kumppaneita saa piestä liikunnaksi asti, robottien sädepistooliammuksia pääsee torjumaan ja tuotantoarvot ovat kohdallaan. Mirage AR seuraa valomiekkaa varsin uskollisesti, joskin sen stereokamerat arvioivat miekan asennon ja lattian korkeuden ajoittain väärin.

Peli paljastaa kuitenkin lisätyn todellisuuden suurimman ongelman: todella toimiakseen sen pitäisi ymmärtää tilaa, ellei tilaksi löydy tyhjää urheilukenttää. Darth Vader seisoo siististi sohvan vieressä vain tuurilla, suuren osan ajasta viholliset leijuvat jossain seinillä ja kalusteiden sisällä. Valoisassa kuva on myös aika haalea, joskin pelattava.

Veikkaan, että Jedi Challenges toimisi paremmin Lenovon tulevalla Mirage Solo -VR-silmikolla, mutta sen taustalla oleva uusi Daydream Standalone -alusta taas ei tue näin edistyneitä liikeohjaimia. Jedi Challenges ja Mirage AR uhkaakin jäädä yksinäiseksi kuriositeetiksi, jonka perusviesti on sama kuin koko tällä artikkelilla: ei vielä.

Lisää todellisuutta

Myös Google Glassin kaltaiset älylasit yrittävät paluuta: Intel Vaunt -projektiolasit, Amazon Alexa -ääniohjattu Vuzix Blade (1 080 euroa, kuvassa alla), Rokid Glass, jossa Rokidin oma ääniohjaus, sekä tusina muuta. Nämä ovat lähinnä infonäyttöjä, eivät saumattomia AR-silmikoita. Niiltä on turha odottaa yhteensopivuutta artikkelin AR-alustojen kanssa.

Kiinnostavampi uutuus ovat Magic Leap -valokenttälasit PC:lle, jotka lupaavat aitoa syvyysvaikutelmaa. Jos joku kävelee lasien ja AR-esineen väliin, esine jää piiloon. Valitettavasti Magic Leap'kin loikkii erillään valtavirran AR-alustoista.

Linkit lisätyn todellisuuden kehitysympäristöihin ja lisätietoihin, sekä mainitut Skrollit 2016.2 ja 2017.1 pdf-muodossa, löydät verkkojatkoiltamme: skrolli.fi/numerot

// Writemonkey ja softasuunnittelun zen

// Writemonkey on ohjelma, jossa ei ole mitään turhaa.
// Siksi siitä voi oppia sen, mikä on olennaisinta.

Teksti: Aleksandr Manzos

Olipa kerran kynä ja paperi. Ne saattoivat olla vähän erilaisia maasta ja aikakaudesta riippuen, mutta toimintaperiaate oli sama. Oli tikku, joka jätti jälkiä ja ohut materiaali, jolle niitä sai jättää. 1800-luvun lopulla käyttöön hiipi mekaaninen kirjoituskone, joka vuosisadan palveltuaan korvautui tietokoneella.

Kirjoitusvälineiden historia ei ole aivan näin yksinkertainen, mutta suuria innovaatiota siinä on niukasti. Tämä ei ole puute, vaan ominaisuus. Kirjoittaminen on aivotyötä, jossa jäsennetään ajatuksia kätevästi kommunikoitavaan muotoon. Jos näppäimistöä ja editointimahdollisuutta ei lasketa, ei ole kovin monta apuvälinettä, jolla tätä työtä voi helpottaa. Tekstinkäsittelyohjelmat on luotu kirjaimellisesti tekstin käsittelyyn. Käsittelyn arvoisten ajatusten loihtiminen on yhtä vaikeaa nyt kuin ennen ajanlaskun alkua.

Tätä taustaa vasten on kiehtovaa, miten paljon eri tekstieditoreja on julkaistu kotitietokoneiden lyhyen olemassaolon aikana. Muotoiluista, oikoluvusta ja muusta on toki valtavasti hyötyä, mutta onko niistä apua töistä raskaimmassa eli ajattelussa? Milloin vähemmän on enemmän? Voiko softa olla zen?

Apinakin osaisi käyttää

Joskus vuoden 2010 tienoilla tekstieditori nimeltä Writemonkey nousi esiin usealla softasivustolla. Kyse oli kuulemma ohjelmasta, joka edisti häiriötöntä kirjoittamista, joten laitoin Wordin hetkeksi syrjään ja otin uuden haastajan kokeiluun. Kokeilu vähän venähti: Writemonkey on ollut jatkuvassa käytössä tähän päivään asti, ja sillä on tullut kirjoitettua noin 300 artikkelin ohella kokonainen kirja.

Slovenialaisen **Iztok Stržinarin** tekstieditori ei ole ainut laatuaan, vaan sillä on sellaisia sisaruksia kuin Dark Room tai yli 40-vuotias Vi. Funktionaalisesti Writemonkey ei eroa suuremmin vaikka Notepadista. Ohjelma tuottaa puhdasta txt-dataa ilman muotoiluja. Selkeimpänä erona teksti tulee Wordin tapaan näkyviin liuskana (mikä helpottaa lukemista), ei pitkinä vaakasuuntaisina jonoina. Keskeisenä psykologisena kikkana Writemonkey peittää ruudun kokonaan, eikä jätä näkyviin edes omia valikkojaan. Kuten softan oma kuvailu kertoo, se ”jättää yksin omien ajatusten ja sanojen kanssa”. On vain teksti ja sen tekijä.

Valikoita kaivamalla paljastuu säätöjä, jolla editorin saa mieleisekseen fontteja ja taustavärejä myöten. Nappia painamalla esiin saa tärkeät tilastot merkkien, sanojen ja muun määrästä. Onpa mukana myös jännempiä ominaisuuksia, kuten kaiken muun paitsi halutun kappaleen piilottava Segment Focus. Näitä voi kuitenkin käyttää juurin niin paljon tai vähän kuin haluaa. Ainoa asia, mitä Writemonkey tyrkyttää, on tyhjä sivu.

Yllättävän kivana lisänä softan voi laittaa näppäimistön matkimaan kirjoituskoneen ääntä. Välittömän palautteen motivaatiovoimaa ei sovi väheksyä. Kuten eräs käyttäjä kehuu Writemonkeyn sivulla: ”Olen yhdistänyt äänen tuottavuuteen ja naputan nopeammin.” Koska softa ei oletuksena käytä oikolukua, naputteluflow ei pysähdy edes pikkuvirheiden nysväilyyn.

Writemonkey ei vaadi asennusta eikä syö resursseja. Näin sitä voi kantaa mukanaan koneesta toiseen vaikka muistitikulla ja aloittaa työskentelyn missä vain, koska vain. Ainoana kauneusvirheenä se on saatavilla vain Windowsille, joskin betatestissä oleva uusi versio on myös omppu- ja pingviiniystävällinen.

Aidattu alue

Ohjelmistopuolen kehitys on mukava juttu, joka laajentaa mahdollisuuksien kenttää. Mutta kuten todettua, on myös syvempää työtä, jossa suurin este ei ole työkalu vaan ihmismieli.

Se mitä Writemonkeylla tehdään oli mahdollista ensimmäisillä kotitietokoneilla, eikä työn perimmäinen luonne ole siitä muuttunut. Päästä kaivellaan ajatuksia ja asetellaan niitä nätisti sanoiksi, jotta joku saa niistä hyötyä ja hupia. Ongelma on, että ajatustyö on väsyttävää puuhaa, josta mieli yrittää lipsua tuon tuosta. Pelkästään tätä kappaletta kirjoittaessa olen joutunut vastustamaan houkutusta tarkistaa netin kuulumiset, keittää kahvit, lopettaa tältä päivältä. Aivot lähettävät jatkuvasti impulsseja huilata! Vaikka pieni tauko

tuntuu ajallisesti mitättömältä, flow-tilan kannalta se voi olla tuhoisa. Transsi, jonka aikana teemme valtaosan merkityksellisestä työstä, on herkkä kuin pieni lintu: vaikea vangita, helppo rikkoa.

Writemonkey minimoi tietokoneella piilevät houkutukset poistamalla kaiken ylimääräisen näkyvistä ja rajaamalla mahdollisuudet tekstin tuottamiseen. Tämä ei estä käyttäjää sulkemasta tai minimoimasta softaa, mutta ainakin työn ja boredpanda.comin välillä on jokin psykologinen este. Ja mitä enemmän pysyy softan asettaman aitauksen sisällä, sitä enemmän treenaa keskittymistään. Jos tekee jotain joka päivä vuosien ajan, se jättää väkisinkin jäljen. Ihmiset ohjelmoivat softia, mutta softat ohjelmoivat myös ihmisiä.

Kaikkia työkaluja ei voi riisua tekstieditorin lailla, mutta häiriöttömyyden periaatetta voi soveltaa aina edes jossain määrin. Digitaiteilijan luonnostelu voi parantua ohjelmalla, joka ei anna leikkiä sadoilla sudeilla, kerroksilla, filttereillä. Muusikko voi löytää sisältään salaisia säveliä, kun käytössä on vain pari alkeellista instrumenttia. Esimerkkinä tästä on varhainen pelimusiikki. Rajat luovat voimaa, joka helpottaa vaikeimman työvaiheen ylittämistä. Sen jälkeen asiat menevät enemmän omalla painollaan. Esimerkiksi kirjoittamisessa editointi on niin paljon luontia kevyempää, että sen voi tehdä millä vain ohjelmalla.

Sakka kupin pohjalla

Writemonkeyn tekijä kuvailee luomustaan zenwareksi. Zen on yliviljelty termi, joka sujautetaan yleensä mukaan, kun kaivataan tujaus henkisyyttä tai ei haluta puhua minimalismista. Onko se sanahelinää, vai voiko softaa käyttää tietyn filosofian ilmaisuun?

Mystisen kuuloinen ”zen” palautuu sanskritin meditaatiota tarkoittavaan sanaan. Zenbuddhalaisuuden ytimessä onkin meditaatioharjoitus zazen, jossa pohjimmiltaan istutaan paikallaan ja annetaan mielen asettautua aloilleen. Vähän kuin juomaa kaataessa sakka laskeutuu itsestään kupin pohjalle, kun sille antaa aikaa. Zazen tähtää todellisuuden kohtaamiseen sellaisena kuin se on, vailla jatkuvaa ajatusten hälinää ja arvottamista. Satori, zenin termi valaistumiselle, tarkoittaakin ei-luokittelevaa mieltä.

Writemonkeyn kaltainen ohjelma kutsuu samoin kohtaamaan oman työtilanteen ilman tekosyitä tai häiriöitä. Tarkoituksena on istua ruudun ääreen ja keskittyä käsillä olevaan tehtävään. Kun mieli alkaa harhailla, sillä on vähemmän houkutuksia, joihin tarttua. Zazenin tavoin se opettaa tuomaan ajatukset takaisin tähän hetkeen. Artikkelin, kirjan tai rakkaudentunnustuksen sepustus on kiinni vain siitä, että välttää editorin sulkemisen ja nakuttaa merkkejä sisään. Buddhalaisuudessa sinne tänne pomppivaa mieltä kutsutaan apinamieleksi. Hauskaa kyllä, kirjoitusapinaksi nimetty softa auttaa sisäisen loikkijan taltuttamisessa.

Kaupallisuus ei kuulu zeniin, anteliaisuus ja jakaminen taas kyllä. Siksi onkin sopivaa, että Writemonkey on jaossa ilmaiseksi, mutta Stržinarille voi lahjoittaa rahaa oman varallisuuden mukaan. Vaikka zenwaressa onkin kyse tarttuvasta tägistä, niin ei se Writemonkeyn kohdalla kovin harhaankaan osu.

Se on sitä mitä se on

Writemonkey herättää vielä yhden kysymyksen: onko kaikkea mahdollista parantaa? Olen käyttänyt samaa tekstieditoria noin seitsemän vuotta, näistä viimeiset kolme samaa versiota (2.7.0.3). Softien esteettistä ilmettä on hyvä päivittää toisinaan, mutta muuten en keksi, mitä muuttaisin Writemonkeyn varsinaisessa toiminnassa. Paperi on paperi ja häiriötön tekstikenttä on häiriötön tekstikenttä.

Writemonkeyn täysin uudistettu kolmosversio on tehnyt pitkään tuloaan ja on tällä hetkellä avoimessa betassa. Jossain vaiheessa varmaan siirryn siihen ihan jo tuulettaakseni aivojani, mutta pikakokeilulla en ihastunut. Esimerkiksi useamman tekstitiedoston pitäminen auki on teoriassa käyttökelpoinen lisä, mutta ei palvele sitä orjallista yhteen asiaan keskittymistä, joka synnyttää parhaan tekstin. Taas: Ihan tällaista armottomuutta ei voi soveltaa koko softasuunnittelun kenttään. Aina voi kuitenkin miettiä, milloin liikaa on flow’n kannalta liikaa ja milloin on saavutettu jotain, mistä kannattaa pitää kiinni.

Zenissä asioihin takertuminen nähdään suurena kärsimyksen lähteenä. Kenties minun kannattaa jossain vaiheessa päästää irti työvälineestä, johon olen niin kiintynyt. Mutta kenties meidän kannattaa myös kollektiivisesti luopua luulosta, että kehitys vie aina vain ja ainoastaan eteenpäin.⊠

1. vim

VIM – EPÄINTUITIIVISEN TEHOKAS EDITORI

Vim on monille ohjelmoijille ja Unix-käyttäjille varsin tuttu tekstieditori, jonka askeettisuudesta voi olla montaa eri mieltä.

Teksti: Kari Lavikka

Ensikosketukseni **Vimiin** lienee samanlainen kuin monilla muilla – jouduin siihen vahingossa. Tarkoituksena oli ehkä muokata crontabia tai commitoida muutokset versionhallintaan. Yrittäessäni kirjoittaa tekstiä käyttäytyi tämä editori aivan holtittomasti. Tekstiä katosi, kursori hyppi mihin sattuu, ja paniikkihan siitä tuli. Eihän ohjelmassa ollut edes valikkoa, ja ainoa keksimäni tapa ohjelman sulkemiseen oli tappaa se toisesta terminaalista käsin.

Vimin opettelu on kuin opettelisi soittamaan kitaraa. Aluksi on opiskeltava joitain perusteita, joiden löytäminen vain kokeilemalla on hankalaa: mitä ovat sointu ja skaala, ja miten ne löydetään vaikkapa G-duurisävellajista. Tämän jälkeen on vain yritettävä soittaa, soittaa lisää, soittaa tarkemmin, opiskella lisää teoriaa vähän kerrallaan, ja soittaa taas lisää.

Myös Vimin opetteluun on käytettävä aikaa, mutta se maksaa varmasti itsensä takaisin. Opettelua ei kannata jättää aivan perusteiden tasolle, sillä tällöin jokin yksinkertaisempi editori, vaikkapa nano, on varmasti tuottavampi valinta.

Mikä Vimissä sitten on niin hienoa? Joillekin se on ehkä laajennettavuus, tai että Vim tai sen innoittaja, alkuperäinen vi-editori, on asennettuna valmiiksi melkein koneessa kuin koneessa. Itselleni nuo tosin ovat melko toissijaisia etuja, sillä en kovin usein edes käytä Vimiä. Sen sijaan olen ottanut Vim-tilan käyttöön kaikissa ohjelmissa, joihin se suinkin on saatavilla. Näitä ovat esimerkiksi **Visual Studio Code** ja jopa **Google Chrome**.

Toimintalogiikka tärkeimpänä erona muihin

Vimin suurin ja tärkein ero muihin editoreihin on nimenomaan sen toimintalogiikka tekstiä editoitaessa – siis se kaikkein epäintuitiivisin mutta tehokkain osuus. Tekstissä liikkumiseen voi käyttää tuttuja nuolinäppäimiä, mutta ne kannattaa unohtaa heti aluksi, sillä matka kirjaimilta nuolinäppäimille on pitkä, hiirestä nyt puhumattakaan. Vimin omat ”nuolinäppäimet”, eli **h j k l**, ovat suoraan kymmensormijärjestelmää käyttävän kirjoittajan sormien alla, eikä kättä tarvitse siirtää mihinkään. Vasen käsi kylläkin joutuu painelemaan Esc-näppäintä tiuhaan, aina kun lisäystilasta haluaa takaisin normaalitilaan, jossa navigointi tapahtuu.

Uudelleen sijoitetut nuolinäppäimet eivät toki ole vielä kovin kova juttu. Vimin tehokkuus syntyy mahdollisuudesta yhdistellä normaalitilassa annettavia komentoja. **w** liikkuu sanan eteenpäin, **dw** leikkaa sanan eteenpäin, **d3t** leikkaa kursorista oikealle kolmanteen pilkkuun, ja niin edelleen. Normaali- ja lisäystilassa annettavia loitsuja voi myös tallentaa makroiksi, jolloin saman muokkauksen voi toistaa myöhemmin haluamassaan paikassa.

Jokseenkin jokainen näppäimistön kirjain tai erikoismerkki tekee jonkin toiminnon Vimissä. Vielä lähes parinkymmenen vuoden käytön jälkeen opin yhä uusia toimintoja, joita pikkuhiljaa yritän omaksua ja ottaa käyttöön liikkuessani pitkin tekstiä tai koodia.

Vimiä ei onneksi tarvitse opetella lukemalla ensin käyttöohjekirjaa. Vimin mukana tuleva vimtutor on kelvollinen opastus Vimin käyttöön, mutta internetistä löytyy paljon hauskempiakin opiskelutapoja, esimerkiksi **Vim Adventures** (vim-adventures.com). Vimiä on myös käsitelty perusteellisemmin Skrollin numerossa 2015.3.

Vim-kokemuksen ei kuitenkaan ole pakko olla äärimmäisen karu. On olemassa myös graafisella käyttöliittymällä varustettuja versioita, joissa on valikko- ja työkalupalkit sekä tuki hiirelle. Tehokäyttäjä ei niitä kuitenkaan juuri kaipaa, sillä kaikki on tehtävissä myös näppäimistöltä. Lähinnä graafisuus tarjoaa paremmat edellytykset esimerkiksi tekstin värjäämiseen, kursivointiin ja alleviivaukseen.

Puhtaimmillaan Vimin olemus on kuitenkin vain tekstiruutu odottamassa käyttäjän näppäinkomentoja. Halutessaan käyttöliittymää voi koristella laajennoksilla, jotka näyttävät vaikkapa sisällysluettelon, versionhallinnan aktiivisen haaran tai muita hienostuneita mutta valinnaisia ominaisuuksia.

Tavataan sanoa, että kymmenentuhannen tunnin harjoittelulla lajissa kuin lajissa voi tulla mestariksi. Onneksi Vim on vain työkalu, ja tunnit sen parissa tulevat täyteen yllättävänkin nopeasti. Tärkeää on aina välillä muistaa kokeilla uusia toimintoja. ☒

Python luikertelee Pokittoon

Lisäpotkua ohjelmoinnin opetteluun

Pokitto on laite ohjelmoinnin ja elektroniikan opettamiseen lapsille. Sen käyttö tehostuu entisestään, kun laitteeseen saadaan korkean tason ohjelmointikieli.

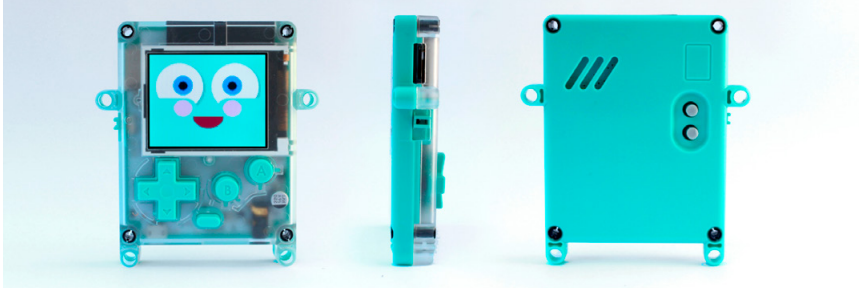
Teksti: Hannu Viitala
Kuvat: Jonne Valola / Pokitto Oy, Hannu Viitala, Maxine Azzara

Pokitto näyttää retropelikonsolilta, mutta sen äly on nykyaikaisista IoT-laitteista ja sulautetuista järjestelmistä tuttua teknologiaa. Lisäksi sen ominaisuuksia voi kasvattaa lähes loputtomasti laajennusportin kautta. Kaikki tämä vaatii tietenkin ohjelmointia. Ohjelmointi helpottuu ja nopeutuu huomattavasti, kun laitteeseen saadaan MicroPython-ohjelmointikieli, joka on suunniteltu sulautettuihin järjestelmiin.

Mikä Pokitto on?

Pokitto on IoT-teknologiaan pohjautuva kannettava, itse koottava peli- ja hyötylaite. Pokitton ”isä” **Jonne Valola** on taustaltaan elektroniikkaa taitava teollinen muotoilija, mikä onkin varsin mielenkiintoinen yhdistelmä.

Muutama vuosi sitten Jonnen lapset osoittivat kiinnostusta elektroniikkaan. Vilkkuvat LED-valot eivät kuitenkaan aluksi sävähdyttäneet tabletteihin ja pelikonsoleihin tottu-



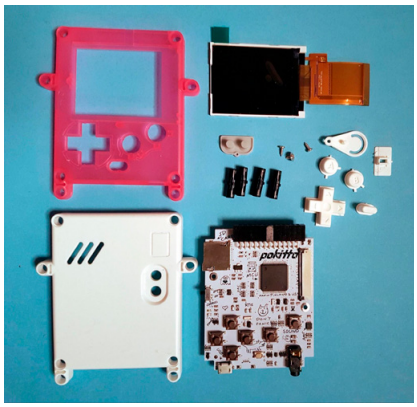
Pelikorttipakan kokoinen Pokitto on koottava avoimen lähdekoodin pelikonsoli, elektroniikan ja ohjelmoinnin opetuslaite sekä IoT-tekniikkaa sisältävä sulautettujen järjestelmien ja elektroniikan ohjausyksikkö.

MCU

- MCU (Micro Controller Unit, mikrokontrolleri) on pieni tietokone, joka sijaitsee yhdellä mikropiirillä.
- MCU = CPU (Central Processing Unit) + muistia + ohjelmoitavia I/O-laitteita (kuten I/O-portti, kello tai sarjaliikennepiiri)
- MCU:ta käytetään yleisesti upotettuna muihin laitteisiin, kuten jääkaappi, mikroaaltouuni tai pesukone. MCU:n sisältävästä laitteesta käytetään suomeksi nimitystä sulautettu järjestelmä.
- MCU vie hyvin vähän virtaa. Siinä missä PC voi käyttää 50 watin edestä tehoa, MCU vie vain 50 milliwattia.
- MCU-mikropiiri on yleensä fyysisiltä mitoiltaan hyvin pieni ja maksaa vain muutaman euron.

POKITTO

- Prosessori: NXP ARM Cortex-M0+, 48 MHz.
- Muisti: 256 kt ROM, 36 kt RAM, 4 kt EEPROM.
- Grafiikka: 2.0" TFT, 220 × 176 pikselin tarkkuus, 65 536 väriä.
- Äänet: 8-bittinen DAC ja 8-bittinen PWM, vahvistin, kaiutin, stereokuulokeliitin.
- Akku: 600 mAh litium-polymeeriakku
- Liitännät: laajennusliitin (GPIO ym.), mikro-USB, microSD.
- Värivaihtoehdot: takakuoren, etukuoren ja nappien väri on valittavissa niin, että tarjolla on yhteensä 336 eri väriyhdistelmää. Ihmettelen, ellei se ole jonkinmoinen ennätys alallaan!
- Kickstarter-kampanjassa halvimman "early bird" -rakennussarjan hinta oli 39 euroa postikuluineen. Nettikaupan myyntihinta on vielä vahvistamatta.



Pokitton osat: etukansi, takakansi, näyttömoduli, kiinnitystapit, ruuvit, näytönpidike, virtakytkin, ristiohjain, napit ja piirilevy akkuineen.

neita nuoria kokeilijoita. Sen sijaan ensimmäinen yhdessä rakennettu pelikonsoli oli yleisömenestys. Näytti siltä, että elektroniikan yhdistäminen pelien ohjelmointiin oli hauska ja motivoiva tapa oppia. Tästä oivalluksesta syntyi Pokitto.

Pokitto rahoitettiin pitkän tuotekehitysvaiheen jälkeen Kickstarter-joukkorahoituksella, jonka Jonne toteutti yhdessä **Daniel DeBeerin** kanssa. Daniel vastaa projektissa laitteen elektroniikan viimeistelystä ja valmistuksesta. Kickstarter tuotti noin 30 000 euroa sekä erittäin aktiivisen yhteisön laitteen ympärille. Pokitto on mikrokontrolleriin (Micro Controller Unit, MCU) perustuva laite, joka sisältää itse MCU-piirilevyn lisäksi LCD-näytön, akun, muistikorttipaikan, USB- ja laajennusportit, kuoret, ristiohjaimen sekä kolme painonappia. Kasattuna laite on suunnilleen pelikorttipakan kokoinen eli hyvin taskuun mahtuva. Pokittosta tekevät mukavan ja sympaattisen näköisen pienet kuoreen muotoillut "kädet ja jalat", joiden tarkoitus on toimia tarvittaessa myös kiinnityskorvakkeina. Ei varmasti ole pelkkä sattuma, että Pokitton kiinnityskorvakkeet ovat juuri sopivia Lego-akselien kiinnitykseen ja että Pokitton mitoitus muutenkin sopii hyvin Lego-palikoiden mittoihin. Laitteesta on aistittavissa huolellista ja pitkälle tulevaisuuteen ulottuvaa suunnittelua.

Pokitton näytössä on 220 × 176 pikseliä ja 65 536 väriä. Audiosta vastaa kaksi itsenäistä 8-bittistä äänipiiriä (PWM ja DAC) joita voi käyttää yhtäaikaisesti, esimerkiksi yhtä musiikin soittamiseen SD-kortilta ja toista ääniefektien luomiseen. MCU:n kellotaajuus on 48 MHz ja muistia siinä on 256 kt flash ROMia, 36 kt RAMia sekä 4 kt EEPROMa. Luit aivan oikein, siis puhutaan kilotavuista, eli muistitila on samaa luokkaa 80-luvun tietokoneiden ja pelilaitteiden kanssa!

Näiden rajoitusten voisi ajatella olevan puute, koska jo peruskännyköistäkin löytyy moninkertaisesti parempia näyttöjä, suurempia muisteja ja tehokkaampia prosessoreita, mutta asiaa kannattaa miettiä toisenkin kerran. Pokitton rajoituksilla on paljon positiivisia vaikutuksia, kuten halpa hinta, pitkä akunkesto, salamannopea käynnistys, yksinkertainen ja itse koottava rakenne, helppo ohjelmoitavuus sekä se, että laite on täysin ohjelmoijan hallittavissa. Koska laitteeseen ei todellakaan mahdu massiivisia ohjelmistorajapintoja tai monimutkaista moniajokäyttöjärjestelmää, ohjelmointi pysyy yksinkertaisena ja ohjelmoijalla on koko ajan selvä käsitys siitä, mitä laitteessa tapahtuu. Yksi innostunut ja taitava pelikoodari voi halutessaan kontrolloida koko laitetta, lähtien grafiikasta ja äänistä ja päätyen ohjelmakoodiin, joka tekee kaiken laitteen käynnistämisestä lähtien ja hyödyntää laitteen resurssit sataprosenttisesti!

Pokitton yläreunassa on laajennusportti, jonka kautta siihen voi lisätä mitä tahansa "hattuja" eli rautalaajennuksia kuten lämpöanturin, IR-kameran, tärinäakun, lisäohjaimen ym. Kuin hatunnostona kultaiselle 80-luvulle GPIO-portin pinnit on liitetty suoraan muistiavaruuteen, joten lisälaitteen saa hallintaan ennen kuin ehtii edes sanoa "poke & peek".

Pokitton käyttämä MCU vastaa muun muassa Googlen IoT-määrityksistä, joten syvällisesti Pokitton toimintaa ymmärtävä käyttäjä saa ikään kuin varaslähdön esineverkkojen täyttämään tulevaisuuteen.

Koodin kehitys Pokittoon

Pokitton järjestelmäohjelmisto on täysin avointa lähdekoodia. Ohjelmoijalla on vapaat kädet muokata sitä tai korvata se vaikka kokonaan uudella. Itse asiassa aina, kun Pokittoon ladataan uusi peli muistikortilta, laitteen flash-muisti ohjelmoidaan uudelleen eli koko ohjelmisto korvataan uudella. Flashaus on yhtä helppoa kuin ohjelman lataaminen muistikortilta tavallisessa kännykässä. Pokitton systeemiohjelmisto huolehtii muun muassa laitteen alustuksesta käynnistyksen aikana, grafiikasta, äänistä ja näppäintapahtumista. Se käyttää vain yhtä prosessia ja säiettä, kuten kaikki muukin koodi Pokittossa.

Peli- ja muu ohjelmistokehitys Pokittolle voidaan tehdä joko simulaattorilla tai itse laitteella, joka on yhdistetty USB-kaapelilla tietokoneeseen. Työkaluina ovat käytössä Code::Blocks, EmBitz ja pilvessä sijaitseva Arm MBed -kehitysympäristö.

Pokitto-simulaattori on avointa lähdekoodia ja toimii Code::Blocksin alaisuudessa niin Windowsissa kuin Linuxissakin. Code::Blocks on ilmai-

nen, nykyaikainen kehitysympäristö ja sisältää muun muassa visuaalisen debuggerin. Simulaattorilla voi kehittää ohjelmia ilman varsinaista Pokittolaitetta, käyttäen C-, C++-, assembler- tai nyt myös Python-kieltä. Yleensä simulaattorissa testattu ohjelma toimii melko hyvin jo sellaisenaan, kun sen kääntää laitteelle ja flashaa Pokittoon. Isompaa tai pienempää säätöä voivat vaatia esimerkiksi sovittaminen laitteen pienemmälle suorituskyvylle, erilainen ruudunpäivitysnopeus sekä direct screen -ominaisuuksien ja laajennusportin käyttö, joista jälkimmäistä ei voi vielä simuloida ollenkaan. Todisteena simulaattorin käyttökelpoisuudesta ja myös yhteisön aktiivisuudesta on se, että Pokittolle oli yhteisön toimesta valmiina tai valmisteilla toistakymmentä peliä ja muuta ohjelmaa jo ennen kuin ensimmäistäkään laitetta oli toimitettu Kickstarter-tilaajille! Nykyään erilaisia pelejä ja demoja on ladattavissa jo kolmisenkymmentä, ja niiden lukumäärä kasvaa Pokitton kotisivulla jatkuvasti, kun yhä uudet koodarit saavat laitteen käsiinsä.

Joko EmBitz- tai Mbed-kehitysympäristöjä voi käyttää ohjelmien tekemiseen Pokitto-laitetta varten. Ensin mainittu on monipuolinen ja tehokas kehitysympäristö, joka asennetaan tietokoneelle. Se tukee muun muassa laitetason debuggausta, joka tosin vaatii lisälaitteen Pokittoon. MBed sen sijaan on hieman rajoitetumpi online-kehitysympäristö, joka ei vaadi minkäänlaista asennusta vaan on käytettävissä mistä tahansa nettiselaimesta.

Ohjelmointia ja elektroniikkaa lapsille

Kun itse lähdin miettimään Pokitton mahdollisuuksia lasten tutustuttamisessa tietotekniikkaan, alkoi nopeasti kirkastua ajatus, että puuttuva lenkki muuten kattavassa systeemissä oli helposti lähestyttävä ohjelmointikieli. Itse olen saanut aikanaan ensikosketuksen ohjelmointiin Basic-kielellä, josta siirryin sujuvasti konekieleen. Python on riittävän helppo kieli oppia, ja sen avulla saataisiin helposti herätettyä lasten ja nuorten mielenkiinto ohjelmointiin.

Peruskouluissa on meneillään paljon puhuttu digiloikka, jonka raameihin Pokitto soveltuu loistavasti. Kuvitellaanpa seuraavanlainen oppitunti peruskoulussa: Luokassa on pöydällä laatikoita, joissa on erilaisia Pokitton osia, kuten näyttöjä, piirilevyjä, erivärisiä kuoria, näppäimiä ja ristiohjaimia. Jokainen ryhmä valitsee osien värit, kerää komponentit ja kokoaa Pokitton alusta loppuun. Sen jälkeen oppilaat saavat valita ”hatun” eli lisämoduulin Pokittoonsa. Niitä on paljon erilaisia: lämpöanturi, kamera, kompassi, kiihtyvyysanturi, jne. Seuraavaksi Pokitto kytketään USB-johdolla tietokoneeseen, jossa interaktiivisen Python-komentotulkin kautta voi kokeilla Pokitton ja hatun ominaisuuksia, kuten lämpötilan lukemista ja kuvan ottamista. Kaikki onnistuu vain muutamalla yksinkertaisella Python-komennolla!

```
# Ladataan kirjastot
import upygame as upg
import urandom as random

# Alustetaan näyttö
upg.display.init()
screen = upg.display.set_mode()

# Luodaan bittikartta, jossa on kaikki oletuspaletin värit (16 kpl)
pokittoPixels = [ 0x01,0x23,0x45,0x67,0x89,0xab,0xcd,0xef ]
surf = upg.surface.Surface( 16, 1, bytearray( pokittoPixels ) )

# Piirtosilmukka
while True:
      # Piirretään näyttöpuskuri täyteen bittikarttoja
      for x in range( 7 ):
            for y in range( 88 ):
                  screen.blit( surf, x*16, y )

      # Piirretään näyttöpuskuri ruudulle
      upg.display.flip()
```

Yksinkertainen MicroPython-ohjelma.

Näin jokainen ryhmä on valmis tekemään pienen Python-ohjelman, joka hyödyntää hatun ominaisuuksia. Valmis Python-koodi siirretään USB-kaapelin tai muistikortin kautta Pokittoon. Nopeimmat oppilaat rakentavat Legoista vielä aiheeseen sopivan telineen tai alustan Pokittolle ja kiinnittävät laitteen siihen. Jokainen ryhmä esittelee luomuksensa muille oppilaille ja opettajalle. Kysymyksiä ja uusia ideoita siitä, mitä kaikkea Pokittolla voisi tehdä, suorastaan satelee ilmassa. Oppitunnin lopuksi opettaja kertoo uusia ja mielenkiintoisia asioita elektroniikasta ja ohjelmoinnista. Seuraavalla oppitunnilla jokainen ryhmä voi joko kehittää ohjelmaansa eteenpäin tai koota uuden Pokitton ja valita siihen jonkin toisen hatun.

Olisitko sinä viihtynyt tällaisella oppitunnilla? Pokittolla tämä on jo nyt täysin toteutettavissa.

Koska olen itse harrastanut paljon pelikoodausta ja se on mielestäni mainio tapa tutustuttaa lapsia ohjelmointiin, päätin siirtää Pokittolle myös osia Pythonin PyGame-pelikirjastosta. Sen kautta voidaan helposti hallita muun muassa Pokitton grafiikkaa, ääntä ja näppäimiä. PyGamen käyttäminen helpottaa myös Python-pelien ja muiden sovellusten siirtämistä eri käyttöjärjestelmistä Pokittolle. Siitä huolimatta peleihin pitää lähes aina tehdä jotain muutoksia, jotta Python-ohjelma toimisi PC:tä rajoitetummassa MCU-laitteessa.

Python ja MicroPython

Python-ohjelmointikieli on varmasti vähintäänkin nimenä tuttu suurimmalle osalle lukijoista, sillä se on kehitetty jo 90-luvun alussa. Kyseessä on siis helposti opittava, tulkattava olio-pohjainen ohjelmointikieli. MicroPython on osajoukko Pythonista. Sen kielioppi ja komennot ovat hyvin pitkälle samat kuin Pythonissa, mutta systeemikirjastot on toteutettu vain osittain. Tämä on luonnollista, koska MicroPython on suunniteltu käytettäväksi sulautetuissa järjestelmissä, joissa on vain vähän muistia ja prosessoritehoa. MicroPythonin on kehittänyt englantilainen teoreettista fysiikkaa, robotiikkaa ja tietokonetekniikkaa opiskellut Damien P. George.

Sulautetuissa ympäristöissä on perinteisesti käytetty ohjelmointikielenä C-, C++- ja assembler-kieliä. Damien halusi selvittää, voisiko niiden sijaan käyttää jotain Pythonin tapaista tulkattavaa korkeamman tason kieltä. Python ei suoraan soveltunut, koska sen muistinkäytön ja prosessoritehon vaatimukset olivat aivan liian suuria. Niinpä Damien teki Pythonin kokonaan uusiksi niin, että sekä ajonaikaista (RAM) että pysyväismuistia (ROM) käytetään niin vähän kuin mahdollista ja että systeemi toimii ilman varsinaista käyttöjärjestelmää. Tästä syntyi avointa lähdekoodia oleva MicroPython-ohjelmointikieli, jota varten hän keräsi noin 100 000 puntaa Kickstarter-kampanjalla.

Kampanjan yhteydessä valmistettiin MCU-pohjainen PyBoard-piirilevy, jota on nyt myyty jo yli 10 000 kappaletta. Yksi merkittävä merkkipaalu saavutettiin, kun BBC:n micro:bit-laite otti MicroPythonin yhdeksi ohjelmointikielekseen. BBC micro:bit on pieni näytötön piirilevy, jossa on valmiina MCU:n lisäksi ledejä, sensoreita, painonappeja, laajennusliitäntöjä, Bluetooth ja USB. BBC-yhtiö on lahjoittanut näitä laitteita Ison-Britannian seitsemäsluokkalaisille noin miljoona kappaletta ohjelmoinnin ja elektroniikan harjoitteluun. MicroPython on saatavilla monille muillekin MCU-pohjaisille piirilevyille sekä muun muassa Raspberry Pi'lle, Windowsille ja Linuxille.

MicroPythonin siirto Pokittolle

Aloitin MicroPythonin siirtämisen Pokittolle "minimal"-projektista, joka on pienin mahdollinen MicroPython-konfiguraatio. Siinä on mukana vain kaikkein välttämättömimmät Python-kirjastot ja -moduulit. Käytännössä porttaus koostui pääosin Pythonin käynnistämisen integroinnista Pokitton startup-koodiin, sopivan MicroPython-konfiguraation luomisesta Pokittoa varten sekä sidontafunktioiden tekemisestä Pythonin ja Pokitton systeemiohjelmiston välillä. Erillinen projekti oli PyGame-siirros, uPyGame, jossa kopioin osia PyGamen Python-skripteistä sekä toteutin Python-luokkia C-kielellä MicroPythonia varten. Tavoitteena oli riittävän hyvän pelintekokirjaston toteutus.

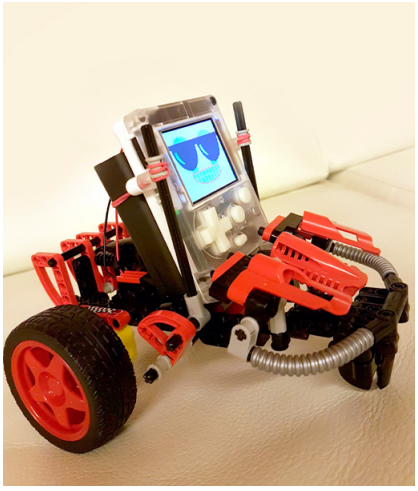
Pokitton 256 kilotavun ROM-muistiin eivät Pokitto-systeemiohjelmiston lisäksi millään mahtuisi edes kaikki Pythonin systeemikirjastot, puhumattakaan lisäkirjastoista, joita löytyy runsaasti eri tarkoituksiin. Koska MicroPython on avointa koodia, on onneksi helppoa tehdä omiin tarpeisiin räätälöity ROM-image ja flashata se Pokittoon. MicroPython on suunniteltu juuri tätä silmällä pitäen ja Pythonin ominaisuuksia ja valittuja kirjastoja onkin helppo muokata konfiguraatiotiedoston avulla. MicroPythonissa on myös käytössä oma mpy-tiedostomuoto, joka on valmiiksi tavukoodiksi käännettyä Python-koodia. Näin koodi vie vähemmän tilaa ROM-imagella. Laitteessa on RAM-muistia vain 36 kilotavua, ja siitä haukkaa oman osansa näyttöpuskuri sekä Pokitton systeemiohjelmiston käyttämä keko- ja pinomuisti (*heap* ja *stack*). Päädyin varaamaan Pythonin käyttöön noin 20 kt kekomuistia, josta siis varataan tilaa dynaamisesti luoduille Python-objekteille ja esimerkiksi interaktiivisen komentotulkkitilan aikana ladatuille Python-ohjelmille. Pythonissa on käytössä roskienkeruumuistinhallinta (*garbage collection*, GC), joka hallitsee kekomuistin käyttöä ja poistaa sieltä tarpeettomia objekteja. Koska RAM-muistia on näin vähän, mahdollisimman suuri osa Python-koodista kannattaa laittaa ROM-imagelle. Jos koodi on vielä käännetty tavukoodiksi, se voidaan suorittaa suoraan ROM-muistista, jolloin RAM-muistia ei kulu itse ohjelmakoodin talletukseen. Myös RAM-muistiin (muistikortilta tai USB-

```
# Ladataan Pin-luokan sisältävä kirjasto.
from umachine import *

# Luodaan Pin-objekti pinniin 14 kirjoittamista varten.
m1 = Pin( Pin.EXT14, Pin.OUT )

# Asetetaan nastan tilaksi "high".
m1.on()
```

Esimerkki GPIO-nastaan kirjoittamisesta esim. ulkoisen DC-moottorin ohjausta varten.



Lego-auton runkoon rakennettu vielä keskeneräinen Pokitto-robottiauto.

portin kautta) ajon aikana ladattu koodi on mahdollista esikääntää tavukoodiksi, jolloin muistia ei vaadita väliaikaisesti siihen, että MicroPythonin täytyisi kääntää koodi ennen sen ajamista. Kekomuistin pirstaloituminen on sekin todellinen ongelma näin pienillä muistimäärillä ja se pitää ottaa huomioon koodia kirjoitettaessa.

Rajoitettua koodausta, rajoittamatonta ongelmanratkaisua

Koodaus rajoitetussa ympäristössä tuottaa yllättäviä haasteita. Tein Python-demon, jossa piirretään ruudulle 100 kertaa 16×16 pikselin kokoinen bittikartta. Se toimi mainiolla 16 kuvan sekuntinopeudella. Tästä innostuneena ajattelin kokeilla 100 bittikartan animointia samaan aikaan ruudulla. Tein periyttämällä PyGamesta oman animoidun sprite-luokan ja aloin luoda instansseja siitä. Homma pysähtyi kuitenkin jo ennen puoltaväliä muistin loppumiseen. Hetken tutkittuani selvisi, että yksi instanssi vie muistia noin 250 tavua jäsenmuuttujien varaaman muistin takia, joten pelkästään spritejä varten olisi pitänyt olla käytettävissä 25 kt kekomuistia!

Toinen esimerkki on peli, jota kehitin aluksi Windowsin Pokitto-simulaattorissa. Siinä piirrettiin joka ruudunpäivityksellä taustakuva ja sen päälle alus. Simulaattorissa kaikki näytti hyvältä ja sujuvalta. Käännettyäni pelin Pokittolle näyttö päivittyi vain noin kerran sekunnissa! Tämän vuoksi muutin toteutusta niin, että taustakuva piirretään vain kerran ohjelman alussa, mutta aluksen liikkuessa sen taakse piirretään ensin riittävän kokoinen pala taustakuvaa (*dirty rectangle* -algoritmi).

Ei pelkästään koodaamista

MCU-pohjaiset laitteet ovat otollista maaperää kaikenlaisille rakenteluprojekteille. Pokittolla se on helppoa, sillä PEX-liitännässä on 26 nastaa, joiden välityksellä lisälaitteet voivat kommunikoida MCU:n kanssa. Suhteellisen pienelläkin elektroniikan tietämyksellä voi muuntaa vaikkapa Raspberry Pi'lle tai Micro Bitille tehtyjä projekteja Pokittolle. Itse olen muuntamassa alun perin Raspberry Pi'lle suunniteltua itseohjautuvaa robottiautoa Pokittolle. Siinä samalla on tarkoitus opettaa lapsilleni elektroniikkaa ja Python-ohjelmointia. Tämän projektin aikana tein MicroPythoniin tuen GPIO-pinnien lukemiseen ja kirjoittamiseen.

Projektini ensimmäinen osa suomenkielisine kurssimateriaaleineen löytyy Pokitton kotisivuilta Wiki-osiosta. Pokitton piirilevyllä käytetään uudempaa 3,3 voltin teknologiaa, joka säästää virtaa. Monissa MCU-rakenteluprojekteissa käytetään kuitenkin vielä 5 voltin jännitettä. Kun näitä projekteja soveltaa Pokittolle, pitääkin olla tarkkana, ettei pinneistä sisääntuleva jännite ylitä 3,3 volttia. Myöskään kaikki GPIO-pinneihin liitettävät ulkoiset komponentit eivät välttämättä toimi Pokittosta saatavalla 3,3 voltin jännitteellä vaan niitä varten tarvitaan esimerkiksi erillinen virtalähde.

Valoisa tulevaisuus

Jonne Valolan mukaan Pokitto laitteena ei suinkaan ole valmis, vaan se tulee kehittymään tulevaisuudessa. Myöskään Pokitton MicroPython-projekti ei missään tapauksessa ole valmis, vaikka se on nyt osa Pokitton virallista kehitysympäristöä. Koodikannassa on valmiina lukuisia ominaisuuksia, joita en ole vielä ehtinyt kokeilemaan tai ottamaan käyttöön, kuten Python-pakettien tukeminen, kuvien ja muiden oheistiedostojen talletus ROMiin ja upotettu assembler-koodi. Koska koodi on avointa, kuka tahansa voi jatkaa työtä ja ottaa käyttöön uusia hienoja ominaisuuksia. Minimissään se voi jopa tarkoittaa vain yhtä pientä muutosta MicroPythonin konfiguraatiotiedostoon. Alkuperäiset tekijät ja yhteisö kehittävät varsinaista MicroPython-projektia edelleen aktiivisesti, ja siihen lisätään yhä uusia ominaisuuksia.

Helppoa ja hauskaa ohjelmointia

Olen vuosia taistellut Unityn ja muiden jättimäisten ohjelmistorajapintojen ja sovelluskehittimien kanssa ja yrittänyt kovalla työllä taivuttaa niitä tekemään, mitä haluan. Pokitto tuo raikkaan tuulahduksen jännittävältä 80-luvulta, höystettynä nykypäivän IoT-yhteensopivalla raudalla ja tulevaisuuden laajennusmahdollisuuksilla. Pokittolle ohjelmien koodaaminen ei ehkä tuota miljoonia euroja, mutta sen kanssa ohjelmointi ei tunnu työltä, vaan se on taas yksinkertaista, helppoa ja hauskaa!

www.pokitto.com
www.pokitto.com/learn/ (Pokitto-wiki)
www.micropython.org
www.pygame.com

Lapset ohjelmoijan opissa
Tarinoita Scratchin ammattilaiselta

Helsingin yliopiston Linkki-resurssikeskus kertoo opettajan perspektiivistä mitä tapahtuu, kun 2000-luvun lapset kohtaavat ohjelmoinnin ensimmäistä kertaa.

Teksti: Ada-Maaria Hyvärinen

Kuvat: Kuvat: Ada-Maaria Hyvärinen, Mitol Meerna, Arto Wikla

Vaara uhkaa! Ohjelmoinnista tulee kansallistaito, kun kaikille opetetaan sitä jo peruskoulussa. Ensin diginatiivit vievät työpaikat, ja seuraavaksi lasten näppäimistöiltä manattu supertekoäly nielaisee meidät kaikki.

Skrollin numerossa 2017.3 esittettiin Scratch-leluohjelmointikieltä. Nyt on aika tarkastella, millaisia nämä Scratchilla aloittavat tulevaisuuden superhakkerit ovat. Tämän jutun kokemukset pohjautuvat ohjelmointikerhoihin, joita järjestää Skrollin kuumin Scratch-kontakti, Helsingin yliopiston Linkki-resurssikeskus. Tutkailtavana ovat keskimääräisten alakoululaisten sijaan valitut lapset, jotka ovat tulleet kerhoihin enemmän tai vähemmän omasta aloitteestaan.

Nämä ohjelmoinnista kiinnostuneet ja siinä mahdollisesti keskimääräistä paremmin pärjäävät lapset eivät kuitenkaan ole mikään homogeeninen ryhmä. Tapoja kiinnostua ohjelmoinnista on yhtä paljon kuin ohjelmoijiakin. Ohjaajan haaste on yrittää tehdä kerhosta sellainen, että suurin osa lapsista saisi opetuksesta jotakin irti.

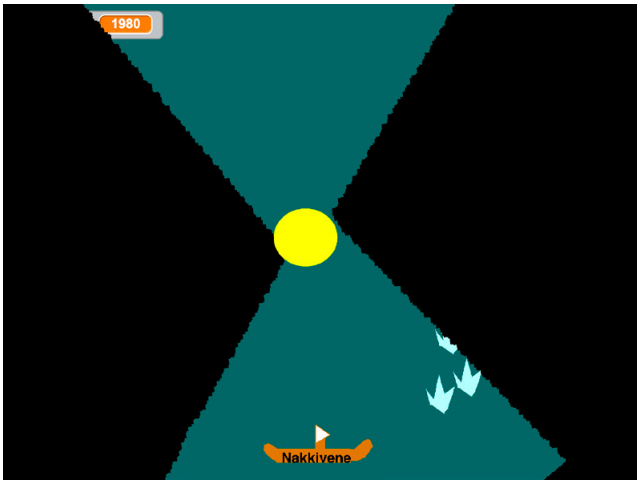
Liian helppoa vai liian vaikeaa?

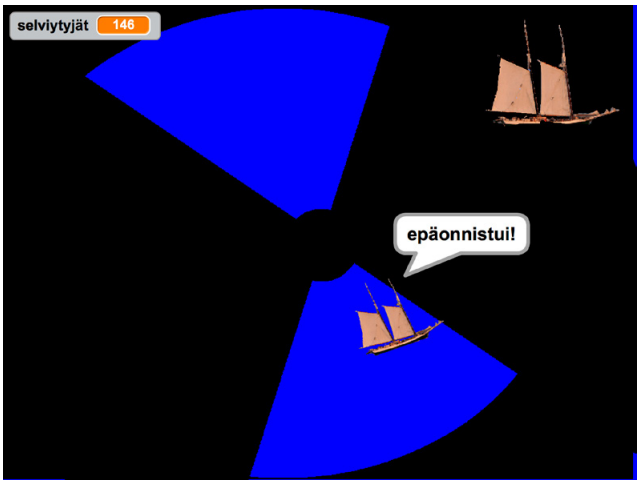
Koska ohjelmointi on käsityötä, koetetaan lapsia kerhossa kannustaa tekemään mahdollisimman paljon itse. Aloittelijoiden kanssa koodia näytetään luokan edessä malliksi, sillä jo Scratch-palikoiden löytäminen ja asetteleminen oikeisiin kohtiin vaatii opettelua. Kerholaisten taitojen kasvaessa siirrytään mallikoodista kirjallisiin ohjeisiin, joita seuraamalla pelin pitäisi syntyä. Lopulta lapsi osaa toivottavasti koodata tarpeeksi, jotta voi tehdä projekteja omien ideoidensa pohjalta.

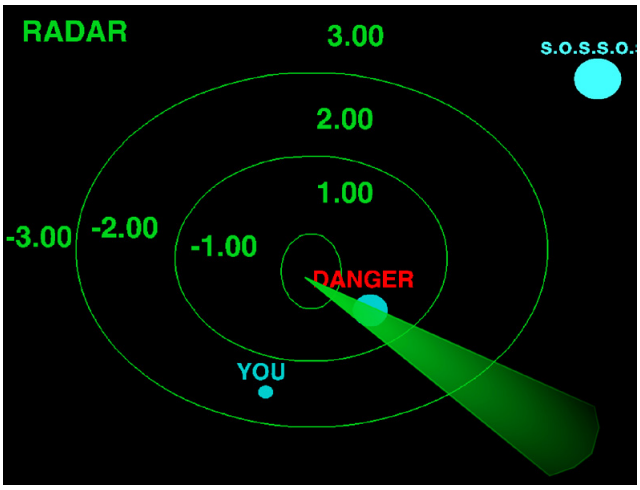
Lapset saavat kysyä apua milloin tahansa, mutta jos kaikki ohjaajat ovat juuri auttamassa jotakuta muuta kerholaista, joutuu omaa vuoroaan odottamaan. Jotkut kerholaiset ilmaisevat avuntarpeensa avoimesti, esimerkiksi: ”Ope hei OPE MÄ TARVIIN APUA. Tää EI TOIMI. Miksei kukaan auta HEI OPE.” Toisessa ääripäässä ovat hiljaiset, jotka eivät tahdo näyttää koodiaan ohjaajalle edes pyydettäessä. Ohjaajaa harmittaa, koska lapsen peli vaikuttaa olevan keskeneräinen ja rikki. Oma toteutus voi kuitenkin tuntua niin henkilökohtaiselta, ettei se kestä ulkopuolisen korjausvinkkejä.

Jos apua ei halua koskaan kysyä, voi edistyminen ja harrastus tyssätä siihen, kun vastaan tulee ongelma, johon ei keksi ratkaisua. Toisaalta ohjelmoijan kannattaa kokeilla ensin itse miettiä pulmiaan ja vasta sitten pyytää apua, koska muuten ei saa tehtyä mitään, jos tukihenkilö ei ole saatavilla.

Tasoerot lasten välillä ovat suuret, ja se vaikuttaa kerhoon, vaikka Scratchia voi oppia kuka tahansa lukutaitoinen. Toiset lapset myös omaksuvat ohjelmointia paljon toisia nopeammin.







Kuvia Linkki-kerholaisten tekemistä meripelastusaiheisista peleistä.

Jo viiden kerhokerran jälkeen toiset lapset käyttävät Scratchia selvästi paremmin kuin toiset, eikä tämä vaikuta korreloivan esimerkiksi iän kanssa. Jos ohjelmointi tuntuu mielekkäältä, sitä tekee mieli tehdä myös kotona, mikä vahvistaa osaamista.

On turhauttavaa, jos opetus etenee liian nopeasti, eikä ohjaaja ehdi tulla heti auttamaan, kun putoaa kelkasta. Sekin on turhauttavaa, jos opetus etenee liian hitaasti, jolloin pitää odotella muita. Hitaammin oppiva viihtyy kerhossa, jos sietää turhautumisen tunnetta ja pystyy odottamaan avunsaantia rauhassa. Nopeasti oppiva pärjää, jos jaksaa odotella muita tai uskaltaa yrittää itse tehdä peliä eteenpäin, vaikka ei olisikaan varma siitä, mitä seuraavaksi tapahtuu.

Kirjalliset ohjeet auttavat etenemistahdin erilaisuuteen, koska silloin lapsi voi käyttää ohjelmointiin juuri sen verran aikaa kuin itse tarvitsee, eikä niin paljon kuin suurin osa ryhmästä tarvitsee. Ala-asteikäinen ei kuitenkaan välttämättä jaksa lukea ohjetta tarpeeksi tarkasti tai pysy kärryillä siitä, onko ohjeessa listatut vaiheet jo toteutettu. Kirjallisten ohjeiden rinnalle Linkissä on tehty joitakin video-ohjeita, koska liikkuva kuva on monille lapsille helpommin lähestyttävä media.

Joillekin kerhoon osallistuville on tärkeää noudattaa ohjeita tarkasti ja tehdä täsmälleen mallin mukainen peli. Moni tarvitsee ohjaajan tukea, koska pelkää koodaavansa jotain väärin. Ohjaajana voi koettaa rohkaista lasta etsimään bugeja itse: ”Kokeile sitä peliä ja jos se tekee mitä haluat, se on varmaan valmis.” Vastaus voi tuntua epätyydyttävältä: mistä voi tietää, ”onks tää nyt oikein”, jos kukaan ei kerro?

Toiset kerholaiset taas eivät pidä ohjeiden seuraamisesta. Jokin oma peli-idea voi viedä mennessään, mikä on tietenkin hyvä asia, jos idea on edes jossain määrin toteuttamiskelpoinen ja lapsi jaksaa siihen keskittyä. Huonommassa tapauksessa omaa ideaa ei ole, mutta myöskään kerho-ohjelma ei jaksa kiinnostaa: ”Tääl kerhossa ei ever tehdä mitään hyvii pelei.”

Lapsi on terve kun se leikkii?

Joskus kuulee sanottavan, että paras tapa oppia on koodiympäristöllä leikkiminen ja satunnainen eri palikoiden kokeilu. Tällöin fokus on itse ohjelmoinnissa, ei valmiissa peliprojektissa.

Hiekkalaatikkomaisuus ei kuitenkaan tunnu juuri lisäävän ymmärrystä siitä, miten koodialueen palikat liittyvät ruudulla pyörivän kissan liikkeisiin ja visuaalisiin efekteihin. Toki väriä vaihtava ja pyörivä kalansilmäkissa on päheä, ja ohjelmointiympäristön kokeilu on hyvä tapa tutustua siihen, mitä Scratchilla voi tehdä helposti.

Lapsia romantisoivissa medioissa tykätään fiilistellä sitä, miten lapsen silmin universumi on maaginen ja käpylehmästä tulee yksisarvinen. Kerhostakin löytyy onneksi tätä lapsitaikaa. Kun lapsi ilmoittaa, että haluaa tehdä Super-Marion tai Minecraftin, ohjaaja huolestuu. Tasohyppelyt ovat Scratchilla hankalia, ja eihän tässä voida vielä mennä mihinkään 3D-juttuihin, kun koordinaatistokin tuottaa vaikeuksia. Lapsen käsitys pelien luonteesta voi kuitenkin olla erilainen. Tyytyväinen kerholainen laittaa rallipeliin punaisen ja vihreän pallon autojen tilalle ja nimeää ne ”Mario” ja ”Luiki”. Pelin nimeksi sopii ”Portalli 2”. Vastaavasti ampumahiihtopelistä saa Minecraft-version laittamalla maalitaulun tilalle lohikäärmeen. Monet haluavat tehdä pelinsä englanniksi kielitaidon puutteesta huolimatta, koska kaikki oikeat pelit ovat englanniksi.

Toisaalta lapsella saattaa olla epärealistinen mielikuva siitä, mitä ohjelmointi on ja mitä Scratchilla saa kerhokerran pituisessa ajassa aikaan. Mistä sen tietäisi, jos ei ole koskaan ohjelmoinut? Ohjaajaa hämmentää, kun aloittelija kysyy ”Miksei tässä näy niitä pisteitä?” vaikka pisteet-muuttujaa ei ole edes luotu. Mutta juuri koodin

ja pelin tapahtumien välisiä yhteyksiä täällä ollaankin opettelemassa.

Kaikkien ei kuitenkaan ole helppo sopeutua siihen, että ohjelmointi on Scratchillakin työlästä ja vaatii opettelua. Tunne on samankaltainen kuin itselle tuli lapsena puuväripaketin hyperrealistisesta muotokuvasta. Ensin innostui. Sitten pettyi, kun itse piirsi kynillä, ja kuvassa näkyi tikku-ukkoja. Samalla tavoin lapsen visio pelistä voi olla oma Final Fantasy tai sitten mystinen Sellainen Ei Tällainen, joka on tosi hieno eikä ainakaan kerho-ohjelmaan kuuluva peli, jossa sammakko loikkii tien yli.

Ohjaaja voi auttaa toteuttamaan lapsen eeppistä peliä niin pitkälle kuin lapsi jaksaa, ja sitten lapsi toivottavasti keksii vähän helpomman idean. Tämä toimii joskus hyvin ja joskus ei ollenkaan, koska se vaatii lapselta pitkäjänteisyyttä sekä kykyä ottaa vieraan aikuisen apua vastaan.

Diginatiivi viihtyy tabletilla

Yleisönosastojen perusteella moni aikuinen pelkää lapsille suunnatun ohjelmoinnin opetuksen koostuvan lähinnä netissä pelaamisesta. Pelko on siinä mielessä aiheellinen, että myös monella lapsella on tämä mielikuva. Kerhossa on sääntönä, että vain itse koodaamiaan pelejä saa pelata, mutta säännön toteutumisen valvominen ei ole aina helppoa. Houkutus pelisivuston avaamiseen ohjaajan silmän välttäessä saattaa käydä ylivoimaiseksi, etenkin jos kerholaisella on kotona kiintiömallinen peliaika. Toiset lapset taas paheksuvat pelaamista kerhossa, sehän on sääntöjen vastaista ja vie aikaa ohjelmoinnilta.

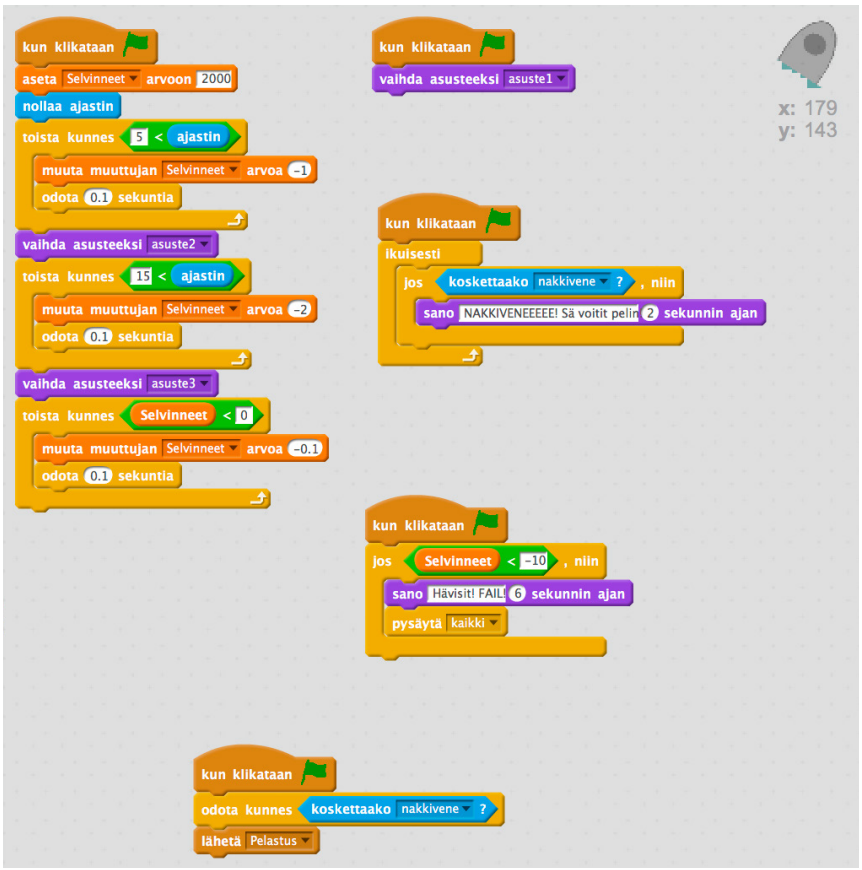
Kun aloitin kerhonohjaajana, hämmennyin siitä, miten huonosti kerhoon valikoituneetkin ala-asteikäiset osaavat käyttää tietokonetta. Melkein jokaisesta aloittelijaryhmästä löytyy joku, joka ei osaa käyttää hiirtä, sillä kännykät ja tabletit tuntuvat olevan yleisempiä lasten työkaluja kouluissa ja kotona. Tämä hankaloittaa opetusta joskus: Scratch turhauttaa, jos hiirellä raahaaminen ei motorisesti suju. Hyvänä puolena ensimmäisten kerhokertojen nettipelaamista hillitsee tehokkaasti se, että monet lapset eivät tiedä, että selaimessa voi olla samaan aikaan useita välilehtiä auki.

Mädättääkö Scratch aivot?

Scratch opetuskielenä herättää aikuisissa ohjelmoijissa tunteita. Ensimmäinen kosketus ohjelmointiin on tärkeä, ja useimmilla ohjelmoijilla on jokin mielipide siitä, mikä on paras aloituskieli. Usein ehdokkaina ovat kieli, jolla ohjelmoija itse aikoinaan aloitti, sekä ohjelmoijan nykyinen suosikkikieli. Scratchiin liittyy usein pelkoja siitä, että lapset saavat vääristyneen kuvan ohjelmoinnista: eihän ohjelmointi ole noin helppoa! Oikeassa ohjelmoinnissa ei ole valmista palikkaa, jolla tutkia, koskettaako kissa omenaa, tai jolla heittää seuraava omena satunnaiseen paikkaan pitkin ruutua. Marraskuisessa ohjelmoinnin opetuksen kansainvälisessä ISSEP-konferenssissa eräs osallistuja esitti myös, että Scratch on lapsille vaarallista: se on liian motivoiva, joten lapset alkavat kuvitella, että kenestä tahansa voi tulla ohjelmoija! En aio ottaa kantaa siihen, minkä ikäisenä ohjelmointi tulisi aloittaa ja millä kielellä. Kumotaan sen sijaan muutama Scratchiin liittyvä käsitys. Ensinnäkään Scratch ei ole aina helppoa: sillä on helppo tehdä tietynlaisia yksinkertaisia pelejä, mutta jos haluaa tehdä mitään muuta, vaatii ohjelmointi pohdintaa koodaustaitoiselta aikuiseltakin.

Lisäksi lapselle ei ole itsestään selvää, millä tavoin ohjelmointikentän palikat vaikuttavat pelin toimintoihin. Ohjelmointilogiikan kanssa saa olla tarkkana, jos ei ole koskaan aiemmin tehnyt sellaista. Vaikka Scratch-palikat ovat suomea, ei lapsi välttämättä ymmärrä niitä kaikkia: esimerkiksi ”muuttuja” on nuorille kerholaisille uusi käsite, joka saattaa vaatia sulattelua.

Pelin toteuttamiseksi ei riitä se, että keksii, millainen peli voisi olla. Pelaajahahmo, pahikset, aarteet ja liikkuvat taustat pitää kaikki erikseen toteuttaa.



Meripelastuspelin koodia.

Kokonaisen projektin tekemisessä on jonkin verran vaivaa, vaikka ohjelmointi itsessään sujuisi helposti. Usein ei suju.

Ala-asteikäinen lapsi ei lue kaikkia näkemiään sanoja samalla tahdilla kuin aikuinen. Vaikka useimmat palikat erottaa värin ja muodon avulla, saattavat palikat mennä myös sekaisin. Yleinen virhe on **aseta muuttujan** [] **arvoksi** <> -palikan sekoittuminen **muuta muuttujan** [] **arvoa** <> -palikkaan. Sitten lapsi ihmettelee, kun pisteet eivät kasva tai hahmo ei liiku. Myös koordinaatiston **x** ja **y** menevät helposti sekaisin. Lisäksi Scratch ei ilmoita järjettömästä koodista mitenkään: laskutoimituksen **kissa** * **6** tulos on **0**. Scratch-ohjelmaa ei voi kaataa.

Kaikesta tästä huolimatta pidän Scratchista. Sillä voi tehdä monimutkaisia juttuja, jos käyttää niihin aikaa ja vaivaa. Scratchissa kopioiminen ja liittäminen on niin työlästä, että lapset koodaavat mieluummin itse, ja oppivat ehkä vahingossa samalla. En kuitenkaan näe juurikaan syitä käyttää Scratchia sellaisten lasten kanssa, joiden kiinnostus pysyy yllä Pythonin tai C:n kanssa. Scratch on hyvä vaihtoehto luovalle pikkuihmiselle, joka ei osaa englantia ja kirjoittaa näppäimistöllä hyvin hitaasti.

Entäs se oikea ohjelmointi?

Kiinnostavaa kyllä, lapset eivät useinkaan pidä Scratchia oikeana ohjelmointina. Mielikuva mustalle ruudulle kertyvästä vihreästä, kryptisestä tekstistä on vahva. Etenkin vanhemmat lapset saattavat pettyä tajutessaan, että kerhon ohjelmointi onkin värikkäitä palikoita lapsellisen näköisellä nettisivulla: ”Ei tääl kerhossa koskaan ohjelmoida. Tääl tehdään vaan Scratchia.”

Toisaalta tekstiohjelmointikerhossa tekstipohjaisen kielen kuten Javan tai Pythonin väkerrys voi tuntua liian kaukaiselta hakkeri-ideaalista. Silloin lapsi saattaa käyttää kerhoaikaa mieluummin ”häksäämiseen”, eli koneelta löytyvien dokumenttien avaamiseen ja muokkaamiseen vailla mitään erityistä päämäärää. Vastapainoksi kerhosta löytyy kyllä niitäkin, jotka edistyvät aikuisen mielestä huikean nopeasti. Tuleehan siitä vähän typertynyt olo, kun ala-asteikäinen kysyy apua Udacityn neuroverkkokurssin ymmärtämiseen. Kun koulussa ei ole vielä kerrottu, mikä on matriisi tai funktio, koneopimisen matematiikkapuoli tuntuu ajoittain haastavalta.

Olen kuullut toisista kerhoista, että tekstiohjelmointipuolella lapset eivät halua tehdä mitään graafista, koska oikea ohjelmointi on tapahtuu terminaalissa. Linkin kerhoissa on ollut päinvastoin: lapsia innostaa graafisten kirjastojen käytön opettelu, koska niillä voi tehdä ”oikean ohjelman”, esimerkiksi ikkunan, jossa voi painaa OK-nappia! Lisäksi merkityksellistä on kuitenkin tuntunut olevan ohjelman teko tyhjästä. Kerholaiset eivät halua käyttää mitään erityisesti kerhoa varten tehtyä pakettia, vaan oppia ohjelmoimaan niin, että voi tehdä graafisen ohjelmansa kokonaan itse vaikka kotikoneella.

Lapsista kuulee joskus idealisoidusti väitettävän, että he eivät ole vielä oppineet ohjelmointiin liittyviä stereotypioita, kuten että pojat ohjelmoisivat tyttöjä paremmin, tai että ohjelmointiin tarvitaan paljon älyä ja matematiikan taitoa. Kerholaisille stereotyyppiset ajatukset ovat kuitenkin tuttuja. Lapsi saattaa esimerkiksi olla vakuuttunut siitä, että on huono ohjelmoimaan, koska ei osaa matematiikkaa – tai toisinpäin. Tähän varmasti vaikuttaa jälleen paljon se, että kerhoon hakeutuvat lapset ovat itse kuulleet aiemmin ohjelmoinnista tai heidän vanhempansa ovat päätelleet lapsen olevan mahdollisesti kiinnostunut ohjelmoinnista. Lapset ovat siis altistuneet ohjelmoinnille jollakin tapaa jo ennen kerhoon tuloa, toisin kuin esimerkiksi koululuokka, jossa osa ei ole ehkä koskaan aiemmin kuullut ohjelmoinnista.

Onks tää nyt oikein?

Kerhon on tarkoitus opettaa lapsille paitsi itse ohjelmointia, myös ohjelmoinnissa tarvittavia taitoja, kuten tehtävän purkamista osiin, tiedonha-

Linkki-keskuksen koordinaattori Jenna Tuominen vetämässä kerhoa.

kua ja sitä, että aina ei tiedä etukäteen oikeaa tapaa toteuttaa jokin haluttu ominaisuus. Osittain tämä näkyy siinä, että ohjaaja ei aina vastaa lapsen kysymykseen suoraan, vaan ratkaisua voi joutua pohtimaan ja kokeilemaan. Aloittelijoiden kanssa ohjaaja useimmiten arvuuttelee vastausta lapselta, mutta etenkin edistyneempien kerhossa lasten kysymykset ja peli-ideat voivat olla varsin hankalia.

Syytän koulua siitä, että yleensä kerhoon tulevat lapset hämmentyvät, kun aikuinen ei tiedä tai kerro vastauksia suoraan. Itsehän tajusin vasta yliopistossa, että vaikka joku olisi matematiikan professori, ei tämä silti osaa kaikkea maailman matematiikkaa suoriltaan. Haluamme kertoa tästä lapsille vähän aikaisemmin sekä kannustaa lapsia luottamaan omaan arvioonsa ohjelman toimimisesta, ilman että tarvitsee kysyä aikuiselta ”onks tää nyt valmis”. Lapsi kuitenkin kokee helposti vastakysymyksen ”onko se sun mielestä valmis?” testinä, johon pitäisi tietää oikea vastaus. Lasta voi myös turhauttaa, kun ohjaaja pimittää vastausta (”etsä vois vaan kertoo miten tää tehdään”) tai kun ohjaaja joutuu kokeilemaan eri tapoja keksiäkseen ratkaisun (”siis voisko tänne tulla joskus joku Scratchin ammattilainen pitämään tätä kerhoa!”).

Lasten opettaminen on palkitsevaa. Lapsissa näkee itsensä ohjelmoijana, hyvässä ja pahassa. Ohjeistaessa ”lue ensin ohjeet huolellisesti” tai ”sulje se pelisivun välilehti, et voi keskittyä ohjelmointiin jos mietit sitä” huomaa toistavansa klassista kaavaa, jossa toisten kuuluu tehdä niin kuin minä sanon, ei niin kuin minä teen. Ilmaisunhaluiset lapset näyttävät ohjelmointiin liittyvät tunteensa aikuisia selkeämmin, ja jo Scratchilla aloittelevat ohjelmoivat heittelehtivät koodarin kahden ääripään välillä: Kun kysyy pöydällä retkottavalta lapselta, tarvitseeko tämä apua pelissä, ja saa vastaukseksi: ”En tarvii. Ei kannata yrittää. Se on toivotonta. Se ei tule koskaan onnistumaan.” Ja silloin, kun neuvoo kirjaimellisesti kädestä pitäen, miten palikoita kuuluu kiinnittää toisiinsa, ja ruudun kissa herää eloon, ja lapsi huudahtaa: ”Jes! Se toimii! Olen paras!”

Kaikki Linkin materiaalit ovat vapaasti käytettävissä: suurin osa löytyy osoitteesta linkki.cs.helsinki.fi, mutta Scratch-tili linkki ja Youtube-kanava Linkki Luokka sisältävät myös sellaisia materiaaleja, joita ei ole vielä linkitetty nettisivuille.

Vinkkejä Scratchin opetukseen

- Opeta lasta, edes vähän. Lapsi ei opi ohjelmoimaan Scratchilla itsestään, vaikka Scratch olisi kuinka helppoa. Ukulelekin on helppo soitin, mutta lapsi ei opi soittamaan sitä, jos lapselle vain lykätään ukulele käteen ja sanotaan, että musiikki on kivaa.
- Opettele ensin itse tekemään Scratchilla edes jotain. Jos et osaa ohjelmoida, etsi joku tutoriaali ja seuraa sitä. Jos osaat ohjelmoida, omaksut Scratchin luultavasti nopeasti. Kannattaa kuitenkin kokeilla ensin ilman lasta.
- Valitse tilanne, jossa lapsella on mahdollisuus jaksaa keskittyä koodaamiseen. Jos lapsi esimerkiksi pelaa mielellään selaimessa, voi off-line-Scratch toimia selainversiota paremmin.
- Laita koneesta äänet pois tai käske lasta käyttämään kuulokkeita, jollet halua kuunnella kissan maukunaa tai muita ääniefektejä.
- Opeta lasta tekemään se projekti, jonka itsekin osaat tehdä. Kannusta lasta valitsemaan itse hahmot peliin ja keksimään laajennusideoita.
- Näytä lapselle, mistä löytää lisää tietoa ohjelmoinnista. Esimerkiksi Linkin ohjeiden tulostaminen voi toimia.
- Ei ehkä kannata neuvoa katsomaan mallia muiden peleistä, vaikka koodinluku olisikin hyvää harjoitusta lapselle. Lapsi jumittuu pelaamaan pelejä eikä jaksa lukea koodia.
- Jos lapsi ei innostu Scratchista, tehkää jotain muuta. Jos lapsi haluaa olla hakkeri, opeta lapselle terminaalin peruskäyttöä. Jos lapsi tahtoo kokeilla tekstipohjaista ohjelmointia, etsi siihen tutoriaali tai opeta itse. Jos lapsi ei jaksa keskittyä kokonaisen pelin verran, tehkää ohjelmointipähkinöitä code.org-sivustolla.

Kryptovaluutat eivät vakuuta

Olen seurannut bitcoinin kurssin kehitystä monta vuotta. Melkein aina olen ollut sitä mieltä, että hinta on liian korkealla, mutta olen ollut väärässä kerta toisensa jälkeen. Mielipiteeni ei ole silti vieläkään muuttunut.

Jarno Niklas Alanko
Kuva: Tuukka Norri

Kuvitelmat bitcoinista jonkinlaisena tulevaisuuden maailmanvaluuttana ovat täysin utopistisia. Valtiot eivät katso sivusta toimettomina, kun keskuspankkien ylläpitämä rahajärjestelmä sivuutetaan. Taloudelle on tärkeää, että esimerkiksi inflaatiota ja korkotasoa voidaan säätää olosuhteiden mukaan, eikä tästä mahdollisuudesta luovuta vähällä. Lisäksi bitcoin tekee verotuksen valvonnasta hyvin hankalaa. Valtiot voivat aivan hyvin kriminalisoida kryptovaluuttojen käytön ja tehdä näin käytöstä vaikeaa ja riskialtista. Kiina on jo ottanut askelia tähän suuntaan. Riittäisi, että Trump naputtelisi yhden kieltämiseen viittaavan twiitin, jotta kurssi lähtisi syöksykierteeseen.

Lisäksi bitcoinissa on hankalia tietoturvaongelmia. Jos joku hakkeroi lompakkosi yksityisen avaimen, niin rahasi ovat oikeasti peruuttamattomasti mennyttä. Vuonna 2014 sen hetken suurin bitcoin-pörssi Mt. Gox joutui konkurssiin hakkeroinnin takia. Tietoturva-aukkoja tulee olemaan niin kauan kuin ihmiset ovat erehtyväisiä, joten miljardien eurojen kokoisen bitcoin-pörssin tietoturvavastaavaa ei todella käy kateeksi.

Jos nyt tietoturvaongelmat sivuutetaan, jotain järkeä on siinä ajatuksessa, että bitcoinista tulisi ”digitaalinen kulta”, johon voisi laittaa säästönsä turvaan pitkällä aikavälillä. Kuten kultaa, bitcoineja on rajallinen määrä, mikä antaa niille arvoa vanhojen postimerkkien tavoin. Tämä on hyödyllistä erityisesti valtioissa, joissa viralliseen valuuttaan ei voi luottaa. Ongelma on vain, että vaikka bitcoineja on rajallisesti, koko bitcoin-järjestelmä on helposti täysin kopioitavissa, koska sen koodi on avointa lähdekoodia.

Mitä jos joku tekee bitcoin-kloonin, jossa on pieniä parannuksia alkuperäiseen verrattuna? Siirtyykö bitcoinin arvo uuteen valuuttaan? Tilanne olisi sama, kuin että löydettäisiin uusi yleinen metalli, joka olisi kaikin puolin kultaa parempi – kullan arvo laskisi varmasti. Bitcoinin ainoa kilpailuvaltti on brändiarvo ja se, että se oli ensimmäinen kryptovaluutta, mutta se ei välttämättä riitä pitkälle.

Kaikesta huolimatta hype käy yhä kuumempana, joskin se hieman laantui alkuvuoden hintaromahduksen myötä. Kun kuulin alkuvuodesta baarin vessassa ohimennen jonkun sanovan laittaneensa 40 tuhatta euroa bitcoiniin, alkoi tuntua siltä, että bitcoin on viimein lyömässä läpi tavallisen kansan tietoisuuteen. Jos internetin keskustelupalstoja on uskominen, lohkoketjuteknologia on vähintään yhtä suuri keksintö kuin itse Internet. Hypen lomassa tuntuu unohtuvan, että suurin osa vaihdosta on pelkkää hintaspekulointia, ja että varsinaista kaupankäyntiä dominoivat todennäköisesti pimeän verkon hämäräkaupat.

Bitcoinin mukana koko kryptoala älysopimuksineen on nosteessa. Miljardeja on kiinni kryptoprojekteissa, joilla ei ole vielä edes toimivaa tuotetta. Hyväuskoiselta viedään rahat alta aikayksikön. Esimerkiksi Bitconnect-niminen ilmiselvä pyramidihuijaus onnistui nousemaan yli miljardin dollarin markkina-arvoon lupaamalla jopa 40 prosentin kuukausituoton, kunnes alkuvuodesta sirkus päättyi ja sijoittajien käteen jäi vain lähes arvottomia Bitconnect-tokeneita.

Järki käteen. Ilmaista lounasta ei edelleenkään ole. Luultavasti kryptokentällä nähdään vielä menestystarinoita, mutta mistä sitä voisi tietää tässä vaiheessa, mitkä tuhansista projekteista tulevat menestymään ja mitkä floppaavat? Valitseminen on kuin löisi vetoa ravihevosen puolesta. Jännitys on taattu, mutta tuotto-odotus hyvin kyseenalainen. Itse en aio sijoittaa mihinkään kryptoprojektiin, vaan sen sijaan jatkan kiroilemista katsomosta, kun vastoin kaikkia odotuksiani hinnat taas moninkertaistuvat.

Commodoren paluu

Uusnelonen

Commodore 64 oli kaikkein aikojen suosituin kotitietokone, jota valmistettiin ja myytiin vuosina 1982–1994 arviolta 13–17 miljoonaa kappaletta. Nyt sitä saa taas uutena – melkein.

Teksti ja kuvat: Janne Sirén

1200 Wilson Drive, West Chester, Pennsylvanian osavaltiossa Yhdysvalloissa, 27. huhtikuuta vuonna 1994 – Commodoren vanhempi rautainsinööri **Dave Haynie** on ottanut videokameran mukaansa töihin yhtiön Yhdysvaltain pääkonttoriin. Seuraavana päivänä puolet henkilökunnasta saisi potkut. Kahden päivän kuluttua Commodore – tuo Suomen tasavallan tietokoneen, Guinnessin ennätysten kirjaan maailman eniten myyneenä tietokoneena päässeen Commodore 64:n valmistaja – olisi konkurssissa.

Videolla, joka julkaistiin sittemmin nimellä *The Deathbed Vigil and other tales of digital angst*, noin viiden minuutin kohdalla, Dave kävelee pimeiden, autioituneiden hallien lävitse. ”Olemme tehtaassa/varastossa. Täällä olivat tuotantolinjat, välitön varasto – kolme tuotantolinjaa”, kertoo Dave. Aikanaan tehdas toimi vuoroissa vuorokauden ympäri, valmistaen tuotteita Pohjois-Amerikan markkinoille. ”Suurin varasto on nyt suljettu kokonaan, koska ihmiset rullaluistelivat siellä.”

Hallit kumisevat tyhjyyttään. Commodore 64:ää kaupattiin vielä, mutta lähinnä pieniä määriä Aasiassa. Commodoren viimeisen hurraan, Amigan, valmistus oli sekin siirtynyt Filippiineille. Kohta Aasiakin hiljenisi. Amigaa valmistettaisiin vielä pieniä määriä Tolousessa, Ranskassa sekä Pennsylvaniassa QuikPak-nimisen yhtiön voimin vuosina 1995–1997, mutta 1200 Wilson Driven tuotantolinjat olivat vaienneet pysyvästi. Pian ne valtaisi omaan käyttöönsä paikallinen Ostos-TV, QVC.

Pennsylvania oli alkujaan Commodore 64:n piirisarjan valmistaneen puolijohdevalmistaja MOS Technologyn kotipaikka. Commodore oli ostanut yhtiön vuonna 1976, kuusi vuotta ennen kuusnelosen julkistamista, ja muuttanut osavaltioon. MOSin vanha tehdas Norristownin kaupungissa oli suljettu vuonna 1992, mutta Commodoren konkurssin jälkeen MOSin johto osti piiriyhtiön takaisin ja käynnisti tuotannon Norristownissa uudelleen. MOS jatkoi vuoteen 2001 nimellä GMT. Viimeinen naula piiriyhtiön arkkuun oli lopulta pitkäaikainen kemikaalivuoto alueen pohjaveteen.

Hävityksen keskelle jäi kuitenkin jotain, joka palaisi kummittelemaan myöhemmin.

Saksalainen sää(s)täjä

Lentoasemantie 1, Vantaa, Suomi, 3. syyskuuta vuonna 1999 – tämän artikkelin kirjoittaja on juuri poiminut saksalaisen Individual Computers -yrityksen (icomp.de) perustajan ja yrittäjän **Jens Schönfeldin** autonsa kyytiin. Jens on saapunut Suomen Amiga-käyttäjät ry:n Saku 99 -tietokonetapahtumaan. Vuonna 1999 Individual Computers ja Jens tunnettiin etenkin Catweasel-lisälaitteestaan (nimetty Catweazlen mukaan), jolla tavanomaiset PC:t pystyivät lukemaan erikoisempien ja vanhempien tietokoneiden levykkeitä.

Jens oli valinnut hotellikseen Cumulus Hotel Olympian Linnanmäen lähellä, koska se oli halvempi kuin ta-

pahtumapaikan viereinen Sokos Hotel Vantaa. Punaisen Honda Civicin kiitäessä – autobaanojen maasta tulleen Jensin mielestä liian lujaa – kohti tuota nykyistä hotelli Cumulus Kalliota Jens valotti edelleen tarkan Saksan markan filosofiaansa: Koko Catweasel-lisälaite perustui käytettyyn piirierään, jonka Jens sai ostettua halvalla. Uudelleenkäytettyjen piirien merkinnät piti siksi peittää häntälogon sisältäneellä tarralla. Jensin kyky hamstrata ja herättää vanhaa rautaa uuteen eloon oli jo tuolloin ilmiömäinen.

Individual Computers keskittyi pitkään nimenomaan Amigaan ja Amigaan liittyvien PC-lisälaitteiden valmistamiseen. Pieni yhtiö on tänä päivänäkin tunnettu niistä. 2000-luvun edetessä Individual Computers sekaantui kuitenkin enenevässä määrin myös Commodore 64:n maailmaan. Alkusysäyksen antoivat uudelleenohjelmoitavat FPGA-mikropiirit, jotka tekivät kovasti tuloaan vuosituhannen vaihteessa ja mahdollistivat kustannustehokkaan laitekehityksen pienissä volyymeissa.

Ensimmäisten joukossa oli C-One-yhteistyöhanke yhdysvaltalaisen **Jeri Ellsworthin** kanssa. C-One oli FPGA-pohjainen uudelleentulkinta Commodore 64:stä ja sen on väitetty olleen ensimmäinen tietokone, joka perustui puhtaasti FPGA-piireihin – siinä ei esimerkiksi ole CPU:ta. C-One-projekti kaatui riitoihin vähäisen tuotannon jälkeen mutta poiki sentään Minimig- ja Turbo Chameleon -hankkeet.

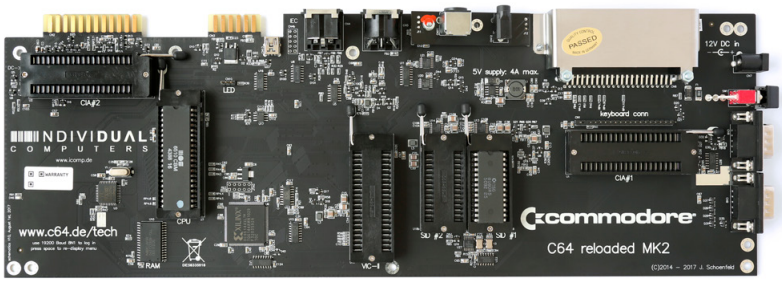
Turbo Chameleon on Commodore 64:n moduuliporttiin sopiva moduuli, joka toimii vaihtoehtoisesti kuusnelosen laajennuksena, simuloiden useita tunnettuja laajennusmoduuleita, tai itsenäisenä FPGA-pohjaisena tietokoneena. Tämän erikoisen laitteen voi siis kytkeä Commodore 64:ään, jolloin se toimii lisälaitteena, tai sen voi kytkeä näyttöön ja näppäimistöön, jolloin se nurinkurisesti toimiikin itse Commodore 64:nä tai muuna ohjelmoitavien piiriensä tukemana tietokoneena.

Retrobuumin laidalla

Individual Computers on kunnostautunut myös muiden Commodore 64 -moduulien ja niiden kuorien valmistajana 2000-luvun alusta alkaen. Uudelleenohjelmoitava Retro Replay -moduuli herätti henkiin vanhat sankarit kuten Action Replay- ja Final Cartridge -laajennusmoduulit. Individual Computers kehitti moduulia varten myös Silversurfer-sarjaportti- ja RR-Net-verkkokortit, joilla Commodore 64:n saa kytkettyä toiseen tietokoneeseen tai ethernet-verkkoon. Retro Replay -moduulilla on nykyisin kevyesti päivitetty seuraaja nimeltään Nordic Replay, joka esiintyy artikkelin kuvituksessa.

Uudelleenohjelmoitavien mikropiirien esiinmarssi 2000-luvun alussa sattui samoihin aikoihin nousevan tietotekniikan retrobuumin kanssa. Esimerkiksi klassinen ja aikanaan supersuosittu Competition Pro -peliohjain oli herätetty henkiin PC-puolella USB-tuotteena. Tarkemmin ottaen Competition Pro Star, tikun jalustassa olevilla lisänäppäimillä. USB-liikenne aiheutti ohjaukseen pientä viivettä, mutta fyysinen toteutus oli uskollinen.

Jens Schönfeldin Individual Computers tarttui tähän markkinarakoon ja osallistui Competition Pron uusintaversion tuottamiseen varustettuna Commodore 64:n ja monien vanhojen kotitietokoneiden käyttämällä 9-pinnisellä niin sanotulla Atari-liittimellä.



Upouusi C64 Reloaded MK2 -emolevy, jossa uusvanhaa varastoa olevat MOS 6510 -suoritin (viikko 46/1986) ja MOS 8580 -äänipiiri (viikko 50/1992) eli SID jo paikoillaan. Kolme tyhjää piirikantaa tarvitsevat vielä käytetyt tai uusvanhat piirit, kakkos-SID ei ole pakollinen. Piirit on helppo lukita ja vapauttaa vivuilla.

USB-tekniikka poistettiin, ja samalla poistuivat ohjauksen viiveet. Automaattitulitus tästä versiosta tosin puuttuu, pieni vipu kytkee vain jalustan lisänäppäimet päälle tai pois. Myös tämä uustuotettu peliohjain esiintyy kuvituksessamme.

Todellinen pommi tuli kuitenkin vuonna 2014, kun Jens Schönfeld ilmoitti alkavansa valmistaa päivitettyä versiota Commodore 64:n 250466-emolevystä. C64 Reloaded -emolevy sopii kaikkiin vanhoihin Commodore 64 -koteloihin ja käyttää keskeisiltä osin alkuperäisiä piirejä. Suurimmat muutokset liittyvät virransaantiin, joka hoidetaan nyt modernilla, tavanomaisella 12 voltin virtalähteellä, ja TV-modulaattoriin, joka on korvattu standardeilla S-Video- ja 3,5 millin audio-liittimillä. Emolevy sai myös reset-tilalla varustetun virtakytkimen. Kevyesti päivitetyt liitännät on suunniteltu niin, ettei koteloon tarvita uusia reikiä.

Toisin kuin esimerkiksi 1541 Ultimaten (1541ultimate.net) kehittelemä lähes täysin FPGA-pohjainen Ultimate-64-emolevy, C64 Reloaded käyttää pitkälti alkuperäisiä piirejä. C64 Reloadedin avulla on siis mahdollista rakentaa alkuperäistä vastaava Commodore 64 uuden emolevyn ympärille. Ensimmäinen tuoteversio tuli laajempaan jakeluun vuonna 2015 ja myytiin nopeasti loppuun.

Uudelleenlataa ja varmista

Seuraaja C64 Reloaded MK2 tuli saataville loppuvuodesta 2017 hienoisin parannuksin: MK2 tunnistaa nyt automaattisesti emolevylle asetetut piirivariaatiot sekä sisältää muun muassa mahdollisuuden käyttää kahta SID-äänipiiriä. Muistit sekä PLA- ja ROM-piirien vastineet ovat valmiiksi MK2-emolevyllä – MK2 ei myöskään vaadi enää EPROM-poltinta, vaan ROMeja voi flashata ja emolevyä konfiguroida kasettiasemaportin

Pixelwizard Retro Shopin 64C-kotelo (oik.) on tarroitukseltaan ja värisävyltään viimeistellympi kuin East Texas Estate Liquidatorin (vas.). Texas-version muovivalu on kuitenkin virheettömämpi.

kulmalle piilotetun huomaamattoman micro-USB-liittimen kautta.

Jensin kyky haalia vanhaa rautaa palkittiin tässäkin: C64 Reloaded MK2:n tilaajalla on mahdollisuus tilata Commodoren vanhoista varastoista peräisin oleva, käyttämätön SID-äänipiiri päiväyksellä 50/1992. Ensipainokseen sai muitakin piirejä suoraan Jensiltä. MK2:n osalta muut vaaditut piirit, MOS 6510/8500-suoritin, kaksi CIA-piiriä sekä VIC-videopiiri, tulee hankkia muualta. 6510-suorittimesta löytää vielä uusvanhaa varastoa helposti esimerkiksi Ebaysta. Muista käytettyjä on saatavilla helposti, ja jokunen uusvanhan varaston kauppiaskin löytynee, kun etsii.

Yksi olennainen C64 Reloadedin parannus on alkuperäisestä virtalähteestä luopuminen, jonka muistetaan sulattaneen jopa muovimattolattioita. Uusine virtalähteineen C64 Reloadedin kerrotaan käyttävän jopa 75 prosenttia vähemmän virtaa kuin alkuperäinen C64, vaikka uudesta virtalähteestä riittää tuplasti enemmän potkua esimerkiksi laajennuksille. C64 Reloadedin kerrotaan silti olevan täysin yhteensopiva esimerkiksi käyttäjäporttiin kytkeytyvien vaihtovirtaa käyttävien lisälaitteiden kanssa.

Virtalähde ei tule paketissa mukana, vaan sellaiseksi kelpaa Reloaded-emolevyjen kanssa mikä tahansa standardilla 12 voltin DC-liittimellä varustettu muuntaja. Liittimen ulkohalkaisijan tulisi olla 5,5 mm, sisähalkaisijan 2,1 tai 2,5 mm ja sisäjohtimen polariteetin positiivinen. Virtalähteen tulee tarjota vähintään yksi ampeeri virtaa, mielellään kaksi. Kuvanlaadun takia 12 voltin ripplen tulisi olla alle 100 mV. Ajoneuvovirtalähteitä ei suositella, sillä niiden jännite on aavistuksen liian korkea.

Individual Computers on itse myynyt virtalähteeksi MC Powerin mallia PS30IBCAK2000E, joita saa myös muista verkkokaupoista muilla nimillä – me käytimme Flypowerin versiota. Koska uusi virtaliitin on Commodoren alkuperäistä pienempi, muun muassa Pixelwizard Retro Shop (shop.pixelwizard.eu) myy 3D-tulostettua peitelevyä kotelon viimeistelyyn.

Kotelomuotit löytyvät

Uudelleenohjelmoitavien piirien lisäksi toinen nouseva teknologia, 3D-tulostaminen, näyttelee siis roolia vanhojen tietokoneiden päivittämisessä. Mutta myös perinteisellä ruiskuvalamisella on edelleen paikkansa. Palataan vielä sinne West Chesteriin, jossa Commodoren operatiivinen päämaja sijaitsi (juridinen pääkonttori oli verosyistä Bahamalla). Commodore 64:ää valmistettiin aikansa West Chesterissä, ja sieltä käsin koordinoitiin myös yhtiön muita valmistusoperaatiota.

Sen jälkeen, kun Commodore hakeutui konkurssiin 29. huhtikuuta 1994, yhtiön omaisuus myytiin velkojen katteeksi ja hajaantui maailmalle. Maailmalle päätyivät myös Commodore 64C:n – vuonna 1986 julkaistun kosmeettisen päivityksen – sekä kone-

mallien 128 ja Plus/4 koteloiden painomuotit. Kaikkea ei kuitenkaan ollut siirretty Aasiaan, vaan yhdet näistä muoteista löysivät tiensä West Chesteristä Texasiin, Longview'n kaupunkiin.

Riippuen siitä, kenen tarinaa kuuntelee, tämä tapahtui hieman jälkeen konkurssin tai kauan ennen. Itse uskon jälkimmäiseen, Texasin Commodore 64C -muotti kun edellyttää erillisiä näppäimistötukia eli on vanhaa mallia. Alalla on myös tavanomaista, että muovitehtaat säilyttävät asiakkaidensa muotteja. Olisi epätodennäköisempää, että muovitehdas olisi jostain syystä hankkinut vanhanaikaiset muotit konkurssipesältä. Myös siitä jotkut kiistelevät, onko näitä muotteja aiemmin käytetty tuotantoon vai ei.

Kiistatonta joka tapauksessa on, että muotit löytyivät vuosikymmenten jälkeen varastosta Texasista ja päätyivät huutokauppaan vuonna 2014. Huutokaupan pitäjän ystävä oli kiinnostunut, niistä kun saisi hienot olohuoneen pöydät. Tietokonehistorian onneksi paikalle sattui pöytänikkarin lisäksi vanhan tavaran kauppamies **Dallas Moore**, joka tunnisti mistä oli kyse ja pelasti nämä kolme satojen kilojen painoista teräsmuottia.

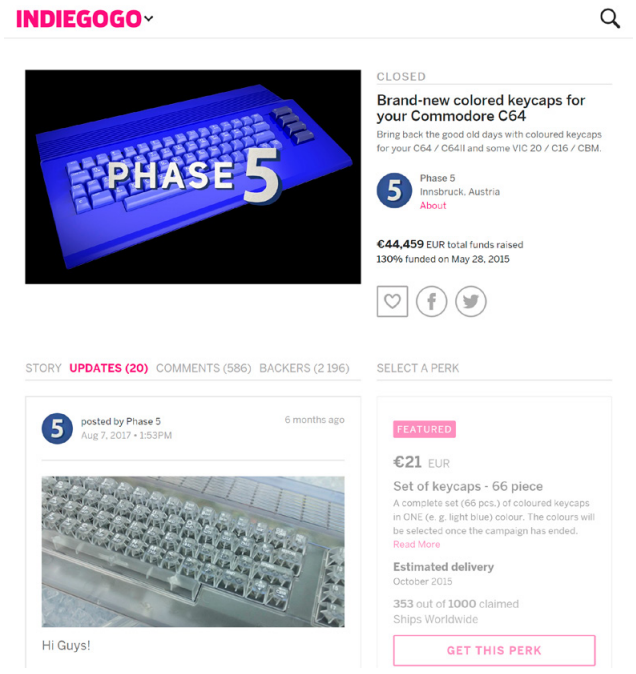
Mielestäni uskottavimman tarinan mukaan Commodore 64C -koteloiden valmistukseen oli 1980-luvulla liittynyt jokin ongelma, jonka vuoksi muotteja oli siirrelty alihankkijalta toiselle ja ne jäivät sille tielleen – lopulta entisen commodorelaisen **E. C. "Bud" Fryen** veljineen omistamalle pienelle Frye International -muovitehtaalle. Frye korjasi ongelman, käytti muotteja aikansa koteloiden valmistamiseen ja sen jälkeen varastoi ne. Moore kuuli kertomansa mukaan tämän tarinan Bud Fryeltä huutokaupan yhteydessä.

Frye International valmisti näppäimistönkuoria muillekin yhtiöille, joskin siirtyi sittemmin tuppervaaratyyliseen muoviastiavalmistukseen. Yhtiö lopetti vuonna 2014 omistajan jäädessä eläkkeelle ja huutokauppasi irtaimistonsa, mukaan lukien Commodoren muotit.

Potkaistaan tyhjää koteloa

Moore ilmoitti löydöstään maailmalle maaliskuussa 2015 Kickstarter-joukkorahoituskampanjan muodossa. Moore lupasi tuottaa erän uusia Com-

Pixelwizard Retro Shopin myymä uustuotettu Commodore 64C -kotelo tulee alkuperäisen tyylisessä pakkauksessa. Myös ohjelappuset kotelolle sekä Individual Computersin myymälle C64 Reloaded MK2 -emolevylle ovat samaan tapaan hienosti Commodore-brändättyjä.



Joukkorahoitus uusille Commodore 64:n näppäinhatuille keräsi rahat nopeasti, mutta tuloksia on tullut hitaasti. Uusin ennuste on, että toimitukset alkavat touko-kesäkuussa. Projektin tavoitteena on Commodoren viimeisimmäksi jäänyt näppäinhattutyyli, jossa kaikki symbolit on painettu näppäinhatun päälle. Värivaihtoehtoja lupaillaan 16 erilaista.

3D-tulostettuja näppäimistötukia ja virtaportin peitelevyjä saa mm. Shapewaysin (kolme vas.) ja Pixelwizard Retro Shopin (oik.) kautta. Tarjolla olevissa malleissa on eroja kaupasta, ostohetkestä ja tulostusmateriaalista riippuen. Lisäksi Pixelwizard on tuomassa myyntiin uudet metalliset näppäimistötuet.

Lähes uusi Commodore 64C – vain näppäimistö ja pari mikropiiriä ovat käytettyjä. Huomaa symbolit näppäinhattujen kyljissä, käytetty näppäimistö on siis vuoden 1986 64C:stä. Myöhemmissä Commodore 64:n näppäimistöversioissa symbolit on painettu näppäinten päälle.

modore 64C -koteloita, jos kiinnostusta olisi. Ja sitähän oli, kymmenkertaisesti minimitavoitteeseen nähden. Rahoitustavoitteeseen päästiin päivässä, ja kuukaudessa kasassa oli lähes 100 000 dollaria. Rahoittajat saivat kotelonsa heinä-elokuussa samana vuonna. Koteloita valmistettiin useissa eri väreissä, mukaan lukien läpinäkyvänä ja sinisenä, minkä vuoksi viime vuosina on näkynyt sangen villin näköisiä kuusnelosia.

Koteloita valmistettiin jonkin verran myös ylimääräisiä, ja beigen väristä versiota on saanut vielä hiljattain Mooren yritykseltä East Texas Estate Liquidatorilta Ebayn kautta. Itse muotit ovat kuitenkin nykyisin Saksassa, sillä Individual Computers osti ne alkuvuodesta 2016. Koteloista tehtiinkin uusi Saksapainos neljässä eri värissä. Tätä erää oli tarkoitus myydä yksinoikeudella Return-lehden verkkokaupassa, mutta viivästysten vuoksi myynti alkoi vasta vuoden 2017 lopulla ja siirtyi Pixelwizard Retro Shopiin. Pixelwizardilta saa uutena myös LED-virtavalon, tarvittaessa useassa eri värissä – Commodore kun käytti monenvärisiä virtavaloja.

East Texasin ja Pixelwizardin koteloversioissa on molemmissa omat vahvuutensa. East Texasin koteloissa on raportoitu vähemmän valuvirheitä kuin Pixelwizardin versioissa, mahdollisesti jälkimmäisessä käytetyn kellastumattoman muovilaadun vuoksi. Tämä saattaa näkyä Saksa-koteloissa etenkin pohjassa ja tummissa väriversioissa. Toisaalta Pixelwizardin versio on lähempänä aitoja värejä ja tulee myös alkuperäistä muistuttavalla Commodore-tarralla varustettuna, johon on lisensoitu logokin.

Omassa vertailussani perusbeiget versiot molemmilta toimittajilta ovat tasaväkiset. Pixelwizardin kotelossa on tosiaan – lähinnä pohjassa – pieniä valuvirheitä, joita East Texasin versiossa ei ole, mutta ne ovat tässä beigessä versiossa lähes näkymättömiä. Pixelwizardin kotelo on päältä katsottuna autenttisempi värisävynsä ja tarransa ansiosta, joten kokonaisuutena se on oma suosikkini. Laatukysymystä kannattaa kuitenkin seurata, jos se tuntuu itselle merkitykselliseltä. Artikkelin verkkojatkoilta (skrolli.fi/numerot) löytyy linkkejä keskusteluun ja kuviin myös tästä.

Lisää vain näppäimistö

Näin näppärällä rakentajalla on jo kasassa lähes täysin uusi Commodore 64C emolevyä, koteloa, virtavaloa ja -lähdettä sekä peliohjaimia myöten. Uustuotettuun kokonaisuuteen jää kuitenkin vielä kirjaimellisesti näppäimistön mentävä aukko. Integroidun näppäimistön koneessa se on melko suuri aukko.

Yksi vaihtoehto on PC-näppäimistö. Retro Innovationsilla (go4retro.com) on C=Key-adapteri PS/2-näppäimistön käytölle kuusnelosessa, ja tee-se-itse-henkisille Githubista löytyy C64-Keyboard-projekti samaan tarkoitukseen. Individual Computers myy jopa C64-näppäimistötarroja PC-näppäimistöille. PC-näppäimistö tuskin kuitenkaan tyydyttää tositarkoituksella harrastavaa, joka on juuri hartaudella rakentanut autenttisen uus-64C:n.

Ensimmäinen askel näppäimistökysymyksen kunnolliseen ratkaisuun on helppo: 3D-tulostuspalvelu Shapewaysin (shapeways.com) kautta saa muun muassa COREi64:n ja Pixelwizardin kehittämiä näppäimistötukia. Pixelwizardilla on vaihtelevasti osia myös omassa varastossa. Niiden avulla käytetyn C64-näppäimistön saa kiinnitettyä uustuotettuun 64C-koteloon. Käytettyjä Commodore 64 -näppäimistöjä kiinnikkeisiin ruuvattavaksi saa muutamilla kympeillä esimerkiksi Ebaysta – toistaiseksi tämä on tarpeellinen välivaihe, uustuotettuja näppäimistöjä kun ei ole saatavilla. Tarvittavia uusia ruuveja ja käytettyjä pienosia myy muun muassa Retro Spares Shop (retroleum.co.uk).

Käytettyjä näppäinhattuja ei kuitenkaan välttämättä tarvita ikuisesti. Itävaltalainen alias **BukoCharly** – taustalla alkujaan tiimi Michael, Peter ja Tom – herätteli klassisen Amiga-piireistä tutun Phase 5 -yritysnimen henkiin vuonna 2015 (phase5.eu) ja keräsi joukkorahoituksella lähes 45 000 euroa Commodore 64:n näppäinhattujen uuteen tuotantoon. Valitettavasti alkuvuoteen 2018 mennessä projekti on käynyt läpi kaikki selitysten seitsemän vaihetta menetetyistä projektijäsenistä kellariin jäätyneisiin pahvilaatikoihin ja etuajassa syntyneisiin vauvoihin. Näppäimistön kulmassa olevan niin sanotun Commodore-näppäimen logon suhteen ollaan myös huovattu edes ja takas lisenssineuvottelujen tilanteen mukaan.

Kirjoitushetkellä projekti on tuottanut runsaasti näppäimistöhattuja ilman merkintöjä sekä joitakin valmiita näytehattuja ja viallisia hattuja, joita on esitetty valokuvissa, messuilla sekä jaettu saksalaiselle C64-yhteisölle. Ilmeisesti viimeisimmät viivästykset ovat liittyneet näppäinmerkintöjen tulostamiseen. Isoin asia eli hattumuotit ja näppäinhatut ovat jo olemassa. Michael on kertonut käyttäneensä jo 50 000 euroa näppäinhattujen ruiskuvalamiseen ja aikovansa saattaa joukkorahoitusprojektin maaliin hinnalla millä hyvänsä – tämän jälkeen näppäinhatut toivottavasti tulevat myös laajempaan myyntiin. Värivaihtoehtoja on lupailtu peräti 16 erilaista.

Täysin uusi Commodore 64 -koteloon sopiva näppäimistö antaa myöskin vielä odottaa itseään. Deskthority.net-käyttäjän tlt:n MeC64-projekti vuodelta 2012 oli pitkään ainoa varteenotettava projekti. Sen ohjeet ja lähdekoodit ovat verkossa, mutta valitettavasti Shapeways on lakannut 3D-tulostamasta MeC64:n tarvitsemia näppäinsovittimia. Breadbox64.com:lla on puolestaan vuonna 2016 aloitettu uudempi näppäimistöhanke, joka on edelleen elossa. Pidetään peukkuja.

Individual Computersilla onkin sitten työn alla jo jotain ihan muuta: Commodore A1200 Reloaded.

Linkit artikkelissa mainittuihin tuotteisiin ja hankkeisiin:
skrolli.fi/numerot

Commodore 64x ja THEC64 Mini

Commodoren mentyä konkurssiin vuonna 1994 yhtiön tavaramerkit ja teknologian oikeudet ovat seilanneet puolivillaiselta omistajalta toiselle ja sirpaloituneet maailmalle. Olemme käsitelleet teemaa Amiga-näkökulmasta muun muassa Skrolleissa 2017.3 ja 2014.3. Kuten Amigankin tapauksessa, myöhempinä vuosina Commodore-nimeä on käytetty pääasiassa nostamaan erilaisten sekundatuotosten profiilia. Amigan tapaan kaikki varsinainen kehitys Commodore-piireissä on ollut kolmansien osapuolten pienyrittäjien ja harrastajien ansiota. Vahvan, vanhan emoyhtiön puuttuessa Nintendon NES Minin kaltaisista laadukkaista tribuuttituotteista on saanut vain unelmoida.

Viime vuosien merkittävin virallinen Commodore-hanke oli vuosina 2010–2012 toiminut Commodore USA. Kyseessä oli uusi, eläköityneen tietoliikenneyrittäjän **Barry Altmanin** Floridaan perustama yhtiö, joka lisensoi Commodore-oikeuksia itselleen. Lyhytikäisen yhtiön "lippulaivatuote" oli Commodore 64x, Intel- ja Nvidia-piireillä ryyditetty Linux-pohjainen tietokone, joka muistutti ulkoisesti (ja vain ulkoisesti, sitäkin löyhästi) ensimmäistä Commodore 64 -leipälaatikkomallia vuosilta 1982–1986. Commodore USA:n tarina päättyi pian Altmanin kuoltua joulukuussa 2012, joskin osia hankkeesta päätyi My Retro Computer Ltd:lle (myretro-computer.com).

Commodore 64x
(Kuva: Commodore USA)

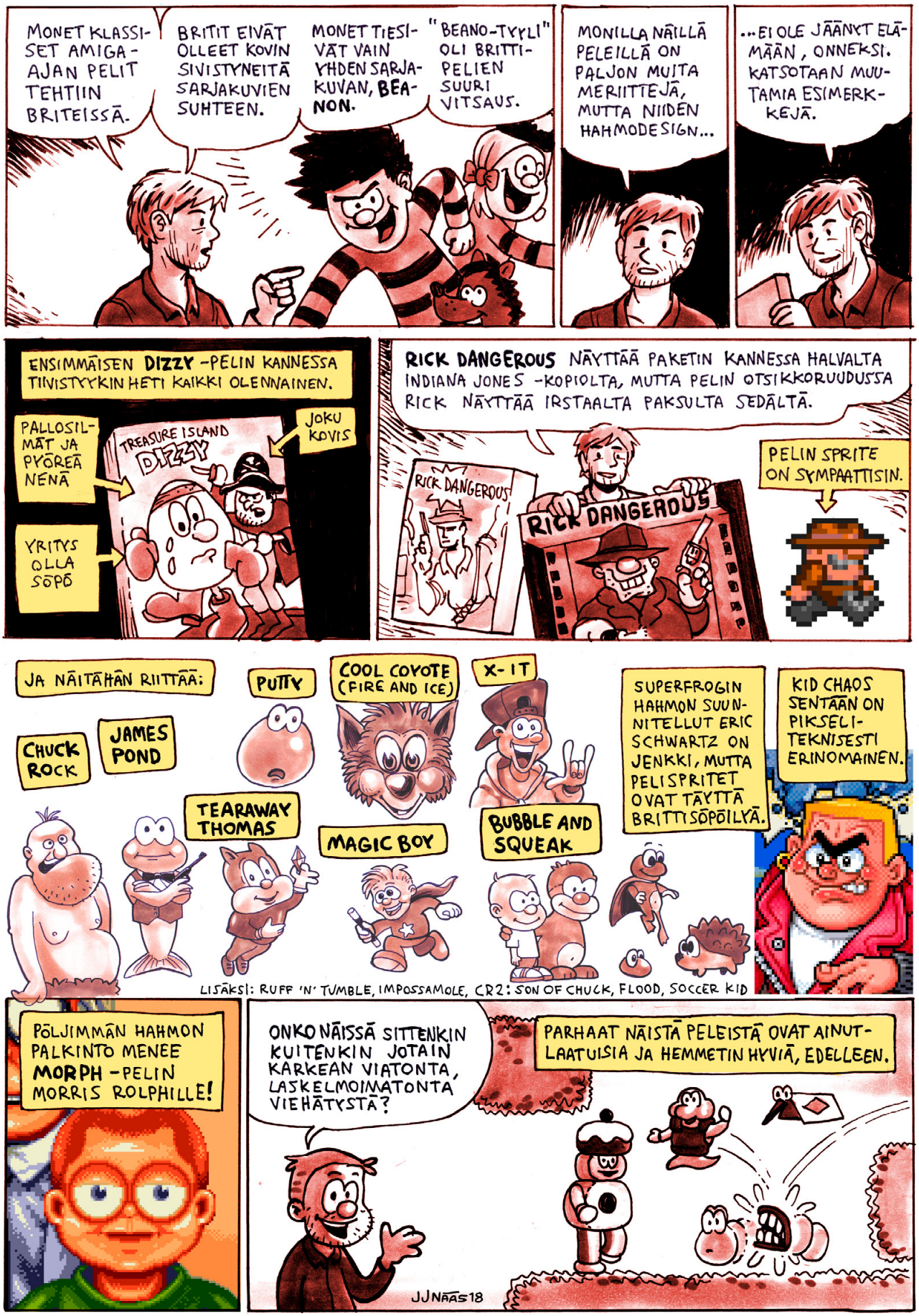
Ajankohtaisempi uutuus on brittiläisen Retro Gamesin THEC64 Mini (thec64.com). Tämä Nintendon NES- ja SNES Mini -menestystarinoiden siivittämä retropeliversio kuusnelosesta pääsi tuotantoon vuoden 2018 alussa. Peliohjaimeen leivottuja, emulaattoriin perustuvia C64-retropelejä on toki julkaistu aikaisemminkin, mutta uutuutena THEC64 Mini on nyt pakattu miniatyyrikokoisen leipälaatikko-C64:n kuoriin. Mukana on 64 lisensoitua klassikkopeliä. Minin näppäimistö ei ole toimiva, mutta ulkoisen näppäimistön kytkemällä pääsee myös basic-ohjelmointikieleen käsiksi.

Myöhemmäksi lupaillaan toimivalla näppäimistöllä varustettua täysikokoista versiota THEC64 Ministä. Kyseessä on sittenkin emulaattoripohjainen järjestelmä, mutta kenties sen näppäimistön voisi vohkia C64 Reloaded -käyttöön. Huomionarvoista tosin on, että Minin nimi ei ole Commodore 64 Mini, tai edes C64 Mini, vaan THEC64 Mini. Myös Commodore-näppäin on korvattu THEC64-näppäimellä. Halvan rahastuksen makua ei täysin voi välttää.

Kaikkea toivoa ei kuitenkaan kannata heittää. Monet varsinaiset Commodore-oikeudet ja -lisenssit ovat nykyisin italialaisen Cloanton ja saksalaisen Individual Computersin käsissä, jotka tunnetaan historiaa kunnioittavista hankkeistaan. Ehkäpä tulevaisuus muistaa Commodorea arvokkaammin. C64 Reloaded ainakin on askel oikeaan suuntaan.

THEC64 Mini
(Kuva: Retro Games/Koch Media)

Morris Rolph



MUISTO VAIN

MEMOTECH MTX500

1980-luvulla Commodorea haastoi useampikin brittikone. Monet tuntevat Sinclairin ja Amstradin, mutta myös Memotechiä myytiin Suomeen.

Teksti: Ismo Utriainen, Janne Sirén
Kuvat: Bilby/Wikimedia Commons, Ismo Utriainen

Memotechin päätuotteet MTX500 ja MTX512 lienevät useimmille tuttuja korkeintaan kuvista, mutta ovat saattaneet jäädä mieleen. Memotechit kun erottuvat maatovereistaan Sinclairista ja Amstradista, sekä muista aikalaisistaan, alumiinikotelollaan. Tämä 2010-luvun perusmetalli oli 80-luvun kotimikrokentässä tuntematon pintamateriaali: kasarilla jopa Macit pakattiin muoviin. Yhtiön toisen perustajan ja pitkäaikaisimman vetäjän **Geoff Boydin** tausta oli metallurgiassa.

Kosketusetäisyydeltä myös täysikokoinen, mekaaninen näppäimistö tuntuu laadukkaalta, etenkin verrattuna kumiseen ja lelumaiseen kilpakumppaniin Sinclair ZX Spectrumiin. Ei ihme, että Memotechin mainoksissa viiletti komea urheiluauto. Memotech oli kotimikrojen Porsche.

Huonomuistinen harvinaisuus

Memotech-nimi liittyy yhtiön alkuvaiheisiin Sinclair-muistilaajennusten valmistajana. Tästä ja 512 kilotavun muistiin vihjaavasta mallinumerosta huolimatta Memotech MTX500 on pienimuistinen kapistus, sillä sitä on vain 32 kilotavua (MTX512-mallissa 64 kilotavua). Tämä oli julkaisuvuonna 1983 kohtuullisesti – 16-bittinen aikakausi oli vielä parin vuoden päässä.

Memotechin pannuhuoneessa on 8-bittiselle kaudelle tyypillistä tekniikkaa: ripeä neljän megahertsin Zilog Z80A -prosessori ja TI:n videopiiri TMS9929, jonka muunnoksia löytyi aikanaan ColecoVisionista Segaan. Massamuistina toimi kasettiasema ja käyttöjärjestelmänä basic. Lisäherkkujakin mukana tuli, kuten Noddy-kieli, eräänlainen HyperCard-hypertekstin ja siten HTML:n edeltäjä.

Koneessa on kaksi Atari-tyyppistä peliohjainporttia, mutta niille ei ole rajapintaa, vaan peliohjaimen liikkeet yhdistyvät näppäimistöön. Kahdella peliohjaimella voi kirjoittaa ruudulle 10 eri merkkiä, joista yksi on välilyönti. Hieman kiusallista silloin, kun vanha peliohjain on löystynyt eikä pysy enää suorassa.

Memotechin vahvuus on laajennettavuus. Siinä on kaksi laajennuskorttipaikkaa, yksi sisäinen ja yksi ulkoinen. Niihin saa sitä lisämuistiakin. Ulkoiseen paikkaan onnekkaat liittivät FDX-lerppuasemaohjaimen, joka paransi koneen käytettävyyttä verrattuna paljon yleisempään kasettiasemaan. Laajennetulla koneella voi ajaa CP/M-käyttöjärjestelmää.

Laajennuksia pystyy myös kytkemään sarjaan, muun muassa lisämuistikortit tukevat usein tätä ominaisuutta. Muita kortteja olivat esimerkiksi token-ring-verkkokortti, sarja- ja rinnakkaisportit sekä Amiga-ohjelmointiympäristöistään tunnetun Hisoftin Pascal-moduuli.

Elämää kuoleman jälkeen

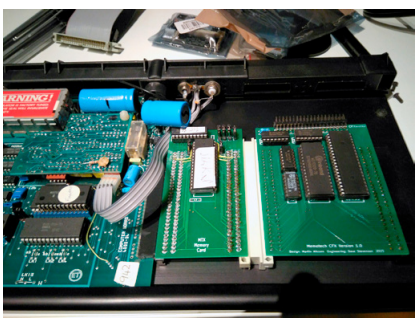
Memotech Ltd:n viimeinen ponnistus oli pyrkiä neuvostomarkkinoille. Yhtiö sijoitti hankkeeseen isosti lainarahaa ja yritystukia. Sopimus kymmenistä tuhansista koneista Neuvostoliiton kouluihin päätyi kuitenkin Yamahalle. Käteen jäi punaisia MTX512-prototyyppejä ja konkurssi vuonna 1986. Sotkusta kuoriutui Memotech Computers Ltd, joka päivitti MTX512:ta hieman ja siirtyi videoseinämarkkinoille, pärjäten siellä 90-luvulle. Useampi lienee nähnyt MTX-pohjaisen (sittemmin PC-pohjaisen) videoseinän, kuin itse kotimikron.

Valmistusmäärä jäi pieneksi: Siinä missä Commodore 64:n realistiset valmistusmääräarviot liikkuvat 13–17 miljoonan hujakoilla, ja Englannin kumista ykköstykkiä ZX Spectrumiakin valmistettiin viisi miljoonaa, Memotech-tietokoneita tuotettiin vain reilut 30 000 kappaletta.

Kotimikro elää vielä harrastajien keskuudessa. Harrastajat ovat tehneet sille laajennuksia, joista yleisin lienee CFX-kortti Compact Flash -massamuistin kytkentään. Koska Memotechin ulkoinen ja sisäinen laajennusväylä on identtinen, CFX käy molempiin. Koneen sisälläkin on hyvin tilaa – ketjutettu lisämuisti ja CFX näkyvät oheisessa sisuskuvassa.

Memotech MTX500 on laatukone, jonka potentiaali jäi hyödyntämättä lyhyen elinkaaren ja vähäisen valmistusmäärän vuoksi. Ohjelma- ja pelimäärä jäi pariin sataan.

Memotech-linkkejä: skrolli.fi/numerot



Matemaattisen taivaanrannan maalarit

FRACTALUS

Rescue on Fractalus! oli varhainen fraktaalien innoittama tietokonepeli. Vuosikymmeniä myöhemmin maisemanluonti kohinalla inspiroi modernin imitaation.

Teksti: Janne Sirén
Kuvat: Janne Sirén, Sakari Leppä, Luke Arnold

Jotkin pelit ovat rakkautta ensisilmäyksellä. Itselleni *Ultimat*, Epyxin *Gamesit*, Sierran *Questit*, mitä näitä nyt on. *Rescue on Fractalus!* (1984) ei ollut. Kun törmäsin peliin ensimmäisen kerran, se oli jo muutaman vuoden vanha. Bongasin sen varmaan alelaarista. Ehkäpä Lucasfilm kannessa oli lupaus jostain suuresta. Todennäköisesti viikkorahat vain olivat lopussa ja se oli halpa.

Vuosi oli noin 1987 ja alusta Commodore 64 – siis suosittu, mutta ei lähellekään se paras *Rescue on Fractalus!* -versio. Pelin kuusneloskäännöstä pidetään yleisesti vähän epäonnistuneena, erityisesti hitaampana kuin kilpailevaa Atarin kasibittiversiota. Kansikuvan dramaattisesti juokseva lentäjä, räjähtelevä toiminta ja tyylikäs ohjaamo eivät nekään suoranaisesti loistaneet bittimuodossaan.

Rescue on Fractalus! ei kuitenkaan unohtunut. Eikä pelkästään siksi, että se oli pelottavin peli heti *Friday the 13th* (1985) jälkeen – oli muutakin.

Fraktaaliajan kasvatti

”KHAAAAAN!” – siinä repliikki, josta elokuva *Star Trek II: Khanin viha* (*The Wrath of Khan*, 1982) varmasti parhaiten muistetaan. Ehkä elokuva kuitenkin ansaitsisi tulla muistetuksi Genesis-aallostaan – erikoistorpedon laukaisun aikaansaamasta ketjureaktiosta, joka maankaltaistaa kuolleen planeetan. Pariminuuttisen kohtauksen puolivälissä tulimerestä nousee jyrkkiä vuoria.

Genesis-aalto, vaikkakin nykymittapuulla karu tehoste, oli aikansa tietokonegrafiikalta saavutus. Se oli ensimmäinen kokonaan tietokoneella luotu kohtaus täyspitkässä elokuvassa. Samoihin aikoihin **George Lucas** heilutteli vielä pienoismalleja kameran edessä *Jedin paluussa* (1983). Lucasilla oli tosin näppinsä myös Genesis-tehosteessa – sen toteutti Lucasin perustama Industrial Light & Magic.

Genesis-vuoret olivat fraktaalivuoria. Fraktaali – tai murtomuoto – on matemaattinen joukko, joka on itsesimilaarinen eli näyttää samanlaiselta tai -kaltaiselta suurennoksesta riippumatta. Kun fraktaalin visualisaatiosta suurentaa mitä tahansa aluetta, siitä löytyy rajattomasti uusia yksityiskohtia, jotka ovat samanlaisia tai samankaltaisia kuin sama joukko muilla mittakaavoilla. Esimerkiksi Julian joukko on kuuluisa fraktaali.

Myös luonnonmaisemien kuten vuoristojen fraktaaliominaisuuksia

on tutkittu. **Benoit Mandelbrot** esitti ensimmäisten joukossa maanmuotojen mallintamista fraktaaleilla. Monet luonnossa esiintyvät ilmiöt ovatkin jossain määrin itsesimilaarisia, maisematkin, ainakin pienessä mittakaavassa. Laajempiin maisemiin vaikuttavat useat erilaiset mekanismit mannerlaatoista veteen, joten niiden mallintaminen edellyttäisi vähintäänkin monien fraktaalien käyttöä.

Fraktaalit hämärsivät geometrisen ja taiteellisen rajaa. Niillä pystyttiin luomaan proseduraalisesti näennäistä satunnaisuutta, jossa oli ihmissilmän odottamaa rytmiä. Tietokonealgoritmilla pystyi viimein visualisoimaan rajattomasti laajenevia, mutta samalla uskottavasti vaihtelevia keinotekoisia luonnonmaisemia – ilman käsityötä. Aikana, jolloin tavanomainen 3D-graafinen vuoristo oli pari symmetristä pyramidia vierekkäin, fraktaalimaisema oli mullistus.

Matemaattinen grafiikkademo

Samana vuonna, kun *Star Trek II* ilmestyi teattereihin, George Lucas perusti Lucasfilm Games -pelistudion elokuvayhtiönsä alaisuuteen. Monet muistavatkin peli-Lucasfilmin kehittämät ja julkaisemat graafiset seikkailupelit, kuten *Maniac Mansion* (1987) ja *The Secret of Monkey Island* (1990). Lucasfilm Games tunnettaisiin myöhemmin nimellä LucasArts, ja se julkaisi videopelejä aina vuoteen 2013 asti.

1980-luvun alussa Lucasfilm Games kehitteli kuitenkin lähinnä toimintapelejä Atarin pelikoneille. Atarin oli määrä toimia pelien julkaisijana. *Rescue on Fractalus!* oli yksi näistä varhaisista peleistä, toinen kahdesta ensimmäisestä. Peli sai alkunsa, kun pari Lucasfilm Gamesin ensimmäistä työntekijää, **David Fox** ja **Loren Carpenter**, jakoivat työhuoneen. Pian he muodostivat pelitiimin.

Carpenter oli ollut kehittämässä Genesis-aaltoa Star Trekiin ja oli fraktaalien asiantuntija. Hän oli esitellyt *Vol Libre* -fraktaalivideotaan SIGGRAPH-konferenssissa jo vuonna 1980 ja diamond-square-algoritmia 1982. Yhdessä **Steve Jobsin** ja **Edwin Catmullin** kanssa Carpenter perustaisi Lucasfilmistä irronneen Pixar-animaatiostudion vuosikymmenen loppupuoliskolla. Vuonna 1983 Carpenter oli kuitenkin piskuisessa Lucasfilm Gamesissa. Yhteisessä työhuoneessaan Fox ja Carpenter päätyivät pohdiskelemaan, voisiko fraktaaleja käyttää mikrotietokoneella.

Rescue on Fractalus! sai alkunsa tästä ideasta ja runkonsa Carpenterin Atari 800:lle kehittämästä fraktaalivuoria piirtävästä grafiikkarutiinista. Peli rakennettiin tämän ympärille. Tässä vaiheessa projektiin liittyi vielä kolmas tekijä, **Charlie Kellner**. Lisäksi pelidivisioonan johtaja **Peter Langston** ja animaattori **Gary Winnick** osallistuivat. Yhtiön toisen pelin *Ballblazerin* (1984) kehittäjä **David Levine** ja myös itse George Lucas auttelivat kulisseissa. Tiimit olivat silloin pieniä ja pelin tekeminen vähemmän muodollista, kuten lukuisista sisäpiirivitseistä voi päätellä.

Pelin tapahtumat esimerkiksi sijoittuivat Tepdi Vad Neroleil Rahcre -tähtijärjestelmään, joka sisältää päätekijöiden nimet väärinpäin. Nimikkoplaneetta Fractalus sai puolestaan nimensä fraktaaleista (engl. fractals). Pelin työnimi tosin oli *Behind Jaggi Lines!*, sekin viittaus grafiikkaan. Pelissä esiintyviä graafisia viivoja ei tuon ajan teknisten rajoitusten vuoksi voitu pehmentää antialiasing-tekniikalla, joten viivat olivat rosoisia (engl. jagged) – nämä jaggiet olivat tietokonegraafikon vihollisia. Näin Jaggi jäi elämään pelissä esiintyvän vihollislajin J'hagga Ri Katchatki lyhennyksenä.

Rescue on Fractalus! suorastaan huokuukin alkuperäänsä grafiikkademona. Pelin tarina sijoitettiin planeetalle, jolla on voimakas kaasuilmakehä, mikä selittää pelissä vallitsevan heikon näkyvyyden. 8-bittisten tietokoneiden tehot eivät riittäneet piirtämään fraktaalivuoristoa kovin kauaksi. Onneksi peli sattui sijoittumaan heikon näkyvyyden planeetalle…

Pelastuspartio Ethercorp

Rescue on Fractaluksessa pelataan fiktiiviseen avaruussotaan osallistuvaa Ethercorp-pilottia, joka lentää pelastustehtäviä kiertoradalla olevalta emoalukselta alla olevalle vuoristoiselle planeetalle. Jaggit ovat asettuneet planeetalle puolustustaisteluun, ripotellen sinne tykkitorneja ja lentäviä lautasia, jotka ovat pudottaneet pelaajan lentäjätovereita. Pelaajan tulee löytää pudonneet lentäjät Valkyrie-hävittäjällään, laskeutua ja tuoda heidät takaisin emoalukselle.

Fractalus-planeetalla on paitsi heikko näkyvyys, myös lentäjien lentopukuja sulattava happoinen ilmakehä. Planeetan yhdeksän minuutin pyörimisnopeus tarkoittaa myös vauhdikkaita vuorokaudenajan vaihteluita myöhemmillä tasoilla, jotka sijoittuvat planeetan päiväntasaajalle. Tehtävällä on siis kiire.

Peli alkaa automaattisella laskeutumisella planeetan ilmakehään. Tämän

Rescue on Fractaluksen (1984, vas.) etumatka uskottavasti kumpuilevien vuorijonojen tuottamisessa oli suuri. MicroProsen hittisimulaattori Gunship (1986, oik.) keinotekoisen näköisine vektoripyramideineen edusti aikakauden tavanomaista tasoa. Kuvat Commodore 64 -versioista.

Fractalus-remaken uuden 0.8-version on tarkoitus ilmestyä tämän lehden julkaisun aikoihin.

jälkeen seurataan tutkan signaalia lentäjiä etsien ja vihollisaseita väistellen. Takaisin voi ampua antimateriakuplatorpedolla. Alun perin tosin ei voinut, David Fox kun oli rauhaa rakastava hippi. Foxin kertoman mukaan George Lucas vaati lisäämään ampumisen testattuaan peliä.

Kun oma lentäjä löytyy, viereen laskeudutaan. Tämän jälkeen lentäjä alkaa juoksemaan kohti alusta ja lopulta koputtaa oveen. Ensin rauhallisesti, sitten kiihtyvällä tahdilla suojapuvun sulaessa planeetan ilmakehässä. Lentäjä tulee päästää sisään oikealla hetkellä ja palata sitten jatkamaan pelastustehtävää. Tähän liittyi myös pelin pelkokerroin: joskus lentäjä paljastuikin Jaggiksi, joka hakkaa tuulilasin rikki ja tuhoaa aluksen. Myös tämä oli George Lucasin idea.

Tarinan mukaan eräs pahaa aavistamaton viisivuotias oli juossut ulos kotoaan kesken pelin, eikä suostunut palaamaan yksin sisään, koska tietokoneessa oli hirviö. Itse olin tuplasti vanhempi ja lähes yhtä järkyttynyt ensimmäisellä kerralla. Jokainen laskeutuminen jännitti sen jälkeen. Itse ”hirviö” oli lähinnä koominen, mutta yllätys mojova. Myöhemmin oppi kyttäämään pään väriä…

Fraktaalien lisäksi Lucasfilmin elokuvatausta näkyi efekteissä. Äänisuunnittelu näytteli pelissä olennaista osaa, mikä oli tuolloin harvinaisempaa. Vain kuuntelemalla pystyi päättelemään oikean hetken päästää lentäjä sisään. Äänet lisäsivät myös kauhua – jos erehdyit päästämään Jaggin sisään, kuulit sen ennen kuin näit.

Pelin räjähdyksetkin olivat erikoisia. Ne matkivat silmän iiriksen sulkeutumista kirkkaiden välähdysten jälkeen, jolloin kaikki vähän pimenee. Charlie Kellner oli animaattori **Mike Jittlovin** suuri fani. Jittlovilla oli tapana käyttää tätä kikkaa stop motion -filmeissään.

Kasibittisten taidonnäyte

Lucasfilmin Atari-yhteistyösopimuksen mukaisesti *Rescue on Fractalus!* kehitettiin Atari 800 -kotitietokoneella, julkaistavaksi moduulilla sille ja Atari 5200 -pelikoneelle vuonna 1984. Kohdealustalla oli näin ollen 1,79 megahertsin 8-bittinen prosessori (yksi syy C64-version hitauteen oli sen 1 megahertsin prosessori). Peli julkaistiinkin Atari 5200:lle jotakuinkin ajallaan. Sitten Atari myytiin hankalan miehen maineessa olleelle liikemies **Jack Tramielille** ja yhteistyö kariutui.

Lucasfilm oli kuitenkin jo ehtinyt lähettää pelin moduuliversion kopiosuojaamattomilla levykkeillä Atarille tarkistettavaksi. Vaikka levykkeistä oli luvattu pitää huolta, *Rescue on Fractalus!* levisi pian piraattipurkkien kautta ympäri maan – vuotta ennen varsinaista julkaisua. Atarin tietokoneversio oli näin ollen kaupallisesti menetetty tapaus, mikä saattaa selittää käännösalustojen runsasta määrää.

Kuriositeettina vuonna 2004 löydettiin prototyyppi myös Atari 7800 -pelikoneversiosta. Vuonna 1984 julkaistavaksi tarkoitetussa 7800:ssa oli MARIA-niminen grafiikkasuoritin, joka muistutti kolikkopelirautaa sadalla spritellään. Siinä oli myös ikäisekseen laaja 256 värin väripaletti. Atari 7800:n *Rescue on Fractalus!* onkin alkuperäisversioista kaunein.

Atari 7800 meinasi myös jäädä Jack Tramielin jalkoihin, päästen markkinoille vasta paria vuotta aiottua myöhemmin. Valitettavasti 7800:n *Rescue on Fractalus!* -prototyypille ei käynyt yhtä hyvin, vaan se jäi keskeneräisenä pöytälaatikkoon. Prototyypistä puuttuu muun muassa ampuminen, laskeutuminen, lentäjien pelastaminen ja useat mittarit.

Monet muut alustat – Commodore 64, ZX Spectrum, Apple II, Amstrad CPC ja Tandy Color Computer 3 – saivat kuitenkin vaihtelevan tasoiset käännöksensä pelistä. Pelille löydettiin uudet jakelijat, jotka tosin edellyttivät moduulien sijaan levykejulkaisua. Julkaisu viivästyi vuodella, koska peliin piti muun muassa lisätä latausanimaatio. Levykeversion muistivaatimukseksi asetettiin 48 kilotavua.

Vallankumouksellisuudestaan huolimatta *Rescue on Fractalus!* ei täysin päässyt irti taustastaan tekniikkademona. Se oli potentiaaliaan suppeam-

Alkuperäisen Rescue on Fractaluksen julkaisi Amerikassa Games-urheilupeleistään tunnettu Epyx, ja Euroopassa julkaisijana toimi Activision. Vasemmalla yhdistetty Commodore 64 / Atari -levykepainos (pelistä oli myös pelkkä Atari-versio). Oikealla C64:n kasettiversio.

Monien painajaisten aiheuttaja: pelastettava lentäjä paljastuikin Jaggi-vihulaiseksi. Ellet toimi nopeasti, Jaggi hakkaa tuulilasin rikki nyrkeillään. Kuva Atari-versiosta.

Jaggista varoitellaan ohjevihkosen lopulla.
Vieressä pelin tekijät ja lucasfilmläiset näyttelevät lentäjiä: Charlie, Loren, David Levine, Peter, Noah Falstein sekä Gary. Projektipäällikkö David Fox juoksee kansikuvassa kypärä päässään.

pi. 1980-luvun vektoripeliskene olisi ollut hyvin erilainen, jos fraktaalimoottorin ympärille olisikin rakennettu vaikkapa MicroProsen massiivinen ja realistinen lentosimulaattori – sisäpiirivitseillä kuorrutetun avaruuskuurupiilon sijaan.

Ajan vektorigraafiset megapelit junnasivatkin vielä vuosia epärealistisen laakeiden peltojen ja symmetristen pyramidivuorten tasolla. Silti, *Rescue on Fractalus!* piirsi itsensä historiankirjoihin rosoisin ja pysyvin kirjaimin. Kun fraktaalimaastoista puhutaan, heti Mandelbrotin jälkeen muistetaan aina Carpenter, *Khanin viha* ja *Rescue on Fractalus!*

Lucasfilmin jakeluyhteistyö Epyxin ja Activisionin kanssa tuotti vielä pari muutakin 3D-toimintapeliä. Vuonna 1985 valmistuneet *Koronis Rift* ja *The Eidolon* jatkojalostivat *Rescue on Fractaluksen* fraktaalitekniikkaa, vaikka eivät varsinaisia jatko-osia olleetkaan.

Toinen tuleminen

Commodore 64:ni vaihtui pian *Rescue on Fractaluksen* jälkeen 16-bittiseen Amigaan. Pelistä ei tehty 16-bittisiä käännöksiä, joten fraktaalit jäivät. Seuraavat vuodet katselimme lentelypeleissä täytettyjä vektoripyramideja kumpuilevien vuorijonojen sijaan. 16-bittiseltä aikakaudella muistan fraktaaligrafiikan lähinnä *Vista Pron* kaltaisista maisemangenerointiohjelmista ja iänikuisisista PD-levykkeiden Mandelbrot-demoista. Fraktaalit olivat edelleen demo.

Tulevina vuosikymmeninä proseduraalinen maisemantuottaminen loikkisi kuitenkin harppauksin eteenpäin, kiitos muun muassa pintakuvioinnin. Eräs toinen SIGGRAPHissa 1980-luvulla vaikuttanut tietelijä oli **Ken Perlin**. Perlinin artikkeli *An image synthesizer* vuodelta 1985 kuvaili Perlin-kohinan käsitteen (Perlin noise), jolla luodaan luonnollisen näköisiä kuvioita tietokonegraafisille pinnoille. Perlin voitti työstään myöhemmin jopa Oscarin.

Erilaisia kohinageneraattoreita on kehitelty sittemmin useita muitakin, ja ne ovat suosittuja välineitä proseduraalisessa maisemantuotannossa. Fraktaali- tai kohina-algoritmeilla lasketulla korkeuskartalla (heightmap) pääsee pitkälle, kuten *Rescue on Fractalus!* osoitti. Toki monimutkaisimmat maisemat ovat yhdistelmiä useista eri tekniikoista – luolat ja kielekkeet edellyttävät muita tekniikoita, koska niihin pelkkä korkeuskartta ei riitä.

Korkeuskarttojen ja kohinan parissa puuhasteli myös australialainen ohjelmistokehittäjä **Luke Arnold** vuonna 2007. Kehittäessään omaa pelimoottoria harrastuspohjalta Arnold kokeili verkosta löytämäänsä korkeuskarttaa, mutta ei ollut tyytyväinen. Hän alkoi tutustumaan proseduraaliseen laskentaan käyttäen kohinaa ja päätti yrittää tuottaa korkeuskartan itse.

”Ensimmäinen maasto, jonka tuotin, oli hyvin rosoinen ja kohinainen. Se muistutti *Rescue on Fractaluksen* vuoria, joten päädyin tekemään pelimoottorin prototyypistäni Fractaluksen näköisen”, kertoisi Arnold sattumuksesta vuosikymmentä myöhemmin Skrollille.

Kolme vuotta kokeilun jälkeen, vuonna 2010, lsdwa.com-sivustolle ilmestyi *Fractalus 0.5.0 alpha*.

Fractalus

Fractalus 0.5.0 alpha oli Luke Arnoldin itsekseen kehittämä, epävirallinen ja maksuton uudelleentulkinta *Rescue on Fractaluksesta* – tai sellaisen runko. Se perustui omaan OpenGL-pelimoottoriinsa. Seuraavan viiden vuoden aikana Arnold julkaisi muutaman isohkon päivityksen, versiot 0.6 ja 0.7, jotka lisäsivät runkoon useita puuttuvia ominaisuuksia. Hiljalleen *Fractaluksesta* kuoriutui oikea peli. Vuoteen 2015 mennessä se sisälsi jo pääasiat alkuperäisestä, joskin maassa juoksevat lentäjät ja viholliset puuttuvat. Pelastettaessa lentäjän tilalla on pelkkä symboli.

Kesällä 2017 projekti aktivoitui taas, nyt ilmoituksella siirtymisestä suositun Unity-pelimoottorin käyttöön. Unity-pohjainen *Fractalus* 0.8 oli työn alla. Arnold perustelee blogissaan, että ”se vähäinen aika mikä projektille oli käytettävissä, kului sekä pelin että pelimoottorin toteuttamisessa – ja yllättäen pelimoottorin kehittäminen olikin varsin aikaa vievää…”

Olin itse ihastellut Arnoldin kädenjälkeä jo vuosien ajan ja pelannut *Fractalusta* enemmän kuin *Rescue on Fractalusta* aikanaan. Kun projekti täytti vuosikymmenen, oli korkea aika ottaa Etherwave-yhteys Australiaan ja kysyä millainen henkilö viettää kymmenen vuotta Fractaluksella?

”Moi! Olen Luke, Perthistä, Länsi-Australiasta”, esittäytyy erittäin ystävällinen Arnold englanniksi ja kertoo jatkaneensa projektia, koska haluaa saada aikaiseksi valmiin pelin soolokehittäjänä. ”Kun aloitin protoilemaan vuonna 2007, pelinkehitys tapahtui tavanomaisesti suurissa tiimeissä. Tämä poikkesi 1980-luvusta, jolloin jotkut pelit kehitti yksi ihminen”, hän jatkaa ja pohtii, oli-

siko tämä edelleen mahdollista. ”Toistaiseksi on ollut, mutta paljon se vaatii aikaa ja vaivaa.”

”Innostuin tietokoneista jo hyvin nuorena, joten vuonna 1986 isäni alkoi opettamaan minulle ohjelmointia Commodore 64:llä. Vuonna 1990 siirryin DOS-pohjaisiin PC-koneisiin ja jatkoin ohjelmoinnin opiskelua, oppien useita kieliä matkan varrella. Vietettyäni enemmän aikaa tietokoneiden ja ohjelmoinnin parissa, ja aloitettuani työt alalla, opin paljon ohjelmakoodin optimoinnista ja grafiikan parissa työskentelystä. Niinpä aloin kehittämään omaa OpenGL/SDL-pelimoottoriani luodakseni puitteet omille peliprototyypeilleni.”

”Pelasin myös monia erilaisia pelejä Commodore 64:llä, Nintendo Entertainment Systemillä ja PC:llä. Monet niistä muistuttivat tyylillisesti toisiaan, oli sivuttais- ja pystyvierityksiä, tasoloikkia, jne. Nautin niistäkin, mutta se mikä todella teki vaikutuksen, oli *Rescue on Fractaluksen* täysi 3D-vuoristo yksityiskohtineen. Pidin muistakin 3D-peleistä, kuten *Mercenary*, *Elite*, *Driller* ja toinen Lucasfilmin peli, *The Eidolon*. Tietokoneen kyky tuottaa reaaliajassa kuvaa 3D-perspektiivissä oli uskomatonta”, jatkaa Arnold.

Tarina on tuttu minulle ja monelle saman ikäpolven harrastajalle. Arnoldin muistot alkuperäisestä *Rescue on Fractaluksesta* eivät suoranaisesti yllätä nekään: ”Kuten uskoakseni useimmilla, yksi vahvimmista muistoistani on ensimmäinen kerta, kun avaruusolento loikkaa aluksen ikkunaan. Se todellakin pelotti lapsena, ja epäröin alkuun pelaamisen jatkamista. Toisaalta pelin hidas vauhti jäi mieleen miinuksena. Arvelen sen olleen vain Commodore 64 -version syytä, sillä vuosia myöhemmin kokeilemani Atari 5200 -versio oli nopeampi.”

Tekniikkademo 2.0

Luke Arnold tiesi, että useista muista vanhoista peleistä oli tehty uusia remake-versioita. Kun kohinalla tuotettu korkeuskartta pelimoottorin raakileessa päätyi muistuttamaan *Fractalusta*, hän tarkasti tilanteen: ”Etsin *Rescue on Fractalus!* -remakea, mutta en löytänyt. Päädyin sitten tekemään pelimoottorini prototyypistä sellaisen.”

Kuten *Rescue on Fractalus!*, *Fractalus*-remake sai siis alkunsa grafiikkademosta. *Rescue on Fractaluksen* keskiössä oli kuitenkin proseduraalisesti luotu fraktaalivuoristo, *Fractaluksessa* näin ei ainakaan vielä ole. Arnold kertoo: ”Haluaisin toteuttaa fraktaaleilla tuotetun maaston, mutta toistaiseksi *Fractalus* käyttää staattista maastoa, jonka olen luonut erikseen. Haluaisin, että peli luo maaston samalla algoritmilla, kuin alkuperäinen, tai sen muunnelmalla, mutta täytyy vielä selvittää, miten alkuperäinen on toteutettu.”

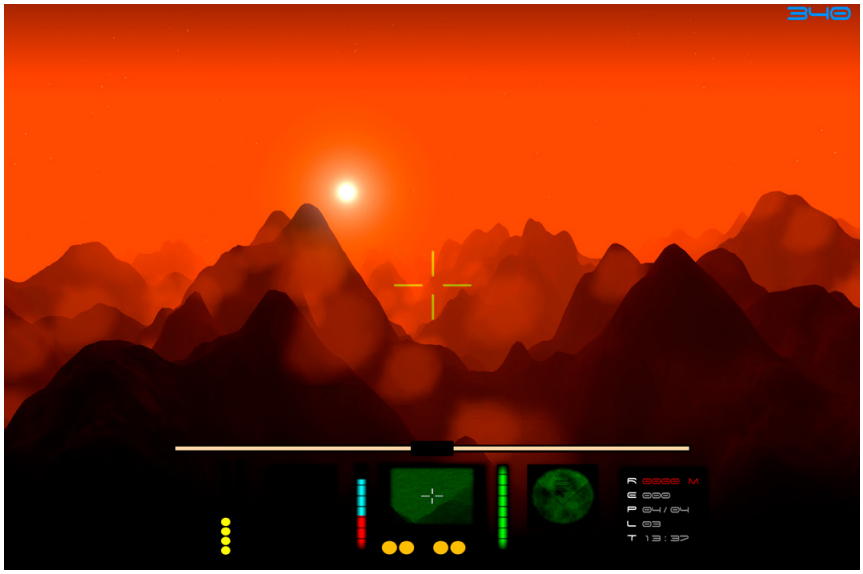
”Kun alun perin protoilin asiaa, en tiennyt paljoakaan satunnaisesta maiseman luomisesta. Loin korkeuskartan GIMPin Create-valikosta löytyvällä Flatland-skriptillä. *Fractalus* käyttää edelleen tätä karttaa. En ole varma Flatlandin algoritmista, mutta sisäisesti se käyttää Solid Noise -toimintoa, jonka tuotos matalimmalla tarkkuustasolla muistuttaa Perlin-kohinaa. Olen sittemmin ohjelmoinut erillisiä kokeiluja myös Perlin-kohinalla.”

Toistaiseksi Fractalus käyttää siis esiluotua maastoa, mutta Arnold alkoo tehdä maisemasta dynaamisen: ”Kun Loren Carpenter teki alkuperäisen *Rescue on Fractaluksen* maisemanluonnin, se on tietysti perustunut hänen fraktaalitutkimukseensa. Sitten kun alan lisäämään ajonaikaista maisemanluontia *Fractalukseen*, aion kokeilla diamond-square-algoritmia, Perlin-kohinaa ja muita. Arvelen, että *Rescue on Fractalus!* saattaa käyttää diamond-square-algoritmia. Odotukseni on, että se näyttää muita enemmän alkuperäisen pelin kaltaiselta.”

Arnoldin motivaatiot ovatkin samankaltaisia kuin monilla harrasteprojekteilla. ”Minulle hauskinta on ollut 3D-peleissä käytettyjen teknologioiden ja tekniikkojen oppiminen, sekä pelisuunnittelun oppiminen tutkimalla *Rescue on Fractalusta* ja muita pelejä. Olen oppinut paljon 3D-renderöinnistä, 3D-mallinnuksesta, peli-, ääni- ja graafisesta suunnittelusta sekä ohjelmoinnista.” Tässä lienee samalla vastaus siihen, miksi harrasteprojekti on kestänyt niinkin pitkään. Matka on tärkeämpi kuin päämäärä. Toki teknologiavalinnoillakin on ollut seurauksensa.

”Tuli myös opittua, että pelimoottorin suunnittelu muuttuu nopeasti kovin monimutkaiseksi ja vaatii enemmän aikaa ja vaivaa, kuin minulla on. Olin kehittänyt oman pelimoottorini protoillakseni uusia ideoita, ja tämä toimikin yksinkertaisille ominaisuuksille, joilla pääsin alkuun. Jatkaessani *Fractaluksen* kehittämistä ymmärsin kuitenkin tarvitsevani kehittyneempiä asioita ja lopulta käytin enemmän aikaa pelimoottorin parissa, kuin itse pelin. Olin tässä vaiheessa jo leikkinyt Unityllä ja huomasin, että siellähän ne tarvitsemani ominaisuudet olisivat valmiina.”

”Fractaluksen porttaaminen Unitylle vapauttaisi pelimoottorin ylläpitämisen taakasta ja saisin keskittyä itse peliin. Tämä onkin sujunut hyvin, joskin kesti hetken oppia Unityn niksit – asiat helpottuivat huomattavasti, kun mal-



Fractalus-planeetan vuorokausirytmi on kiivas. Lyhyt päivä lopussa ja lentäjät kyydissä, aika palata emoalukselle... Vanhempaa, omaa pelimoottoriaan käyttävää Fractalus 0.7:ää (kuvassa) voi pelata myös ilman ohjaamografiikkaa.

Luke Arnold aikoo viedä Fractalus-remakensa maaliin Unity-pelimoottoria käyttäen. Kuvassa versio 0.8 valmistumassa. Uusi Unity-pohjainen versio onkin huomattavasti viimeistellympi kuin aikaisemmat julkaisut.

toin käydä lävitse muutamia heidän opetusprojektejaan. Jahka opin miten Unity haluaa asiat esitettävän, oman pelimoottorini konseptien siirtäminen sinne oli varsin suoraviivaista."

Tavoitteena on, että Unity-pohjainen *Fractalus* 0.8 vastaa ominaisuuksiltaan omaa pelimoottoriaan käyttävää *Fractalus* 0.7:aa. Tämä työ oli haastattelu-hetkellä melkein valmis. Pääsin pelaamaan *Fractalus* 0.8:n kehitysversiota alkuvuodesta, ja Arnold kaavailee 0.8:n julkaisua suunnilleen tämän Skrollin ilmestymishetken tienoille.

Fractaluksen tulevaisuus

Kysyin Lukelta, tuleeko *Fractaluksesta* versiota 1.0 tai 2.0? "Tavoitteeni on, että 1.0 sisältää kaikki alkuperäisen pelin ominaisuudet, joten kyllä, 1.0 on tulossa. Minulla on joitakin ideoita pelin laajentamiselle tämän jälkeen, ja monet ovat lähettäneet minulle ehdotuksiaan, mutta en tiedä vielä mitä tapahtuu 1.0:n jälkeen, jos mitään. Minulla on useita muitakin projekteja, joita työstän ajasta aikaan, liittyen ammattiini. Joitakin hyöty- ja joitakin peliprojekteja, jotka eivät ole vielä julkisia. Haluaisin kehittää myös niitä, mutta ensiksi haluan saada *Fractaluksen* valmiiksi."

Arnold ei esimerkiksi ole poissulkenut ajoittain toivottua keinotodellisuutta: "Unity mahdollistaa tämän, joskin pelimaailman nykyinen piirtämistapa aiheuttaa muutamia haasteita. Minulla olisi kyllä Oculus Rift DK1 testaamiseen."

Entäpä ne juoksevat lentäjät ja Jaggit? "Nykyinen symboli perustuu alkuperäisen pelin lentäjäsymboliin ja pitää paikkaa, kunnes saan toteutettua lentäjän ja avaruusolennon grafiikat. Aion toteuttaa molemmat 3D-malleina ja opettelen parhaillaan mallinnusta Blenderillä. Tällä tahdillani veikkaan, että lentäjä tulee versioon 0.9 ja avaruusolento versioon 1.0."

Mielestäni Arnold on joka tapauksessa jo onnistunut. *Fractaluksen* foorumilla ja some-kanavissa on ansaitusti ihastuneita kommentteja. Toisin kuin alkuperäinen *Rescue on Fractalus*, myös minulle *Fractalus* oli rakkautta ensisilmäyksellä. Uusi versio on huikean tunnelmallinen. Siitä toki myös alkuperäinen ansaitsee kiitoksensa, osa tunnelmasta on sen historiassa. Alkuperäisen *Rescue on Fractaluksen* tärkein saavutus silti on siinä, miten sen käyttämä tekniikka on sittemmin viitoittanut tietä lukuisille muillekin 3D-peleille. Pyramideja löytyy enää Egyptistä.

Lucasfilmin peliosasto lopetti pelituotannon vuonna 2013 Disneyn ostettua yhtiön edellisenä vuonna. 150 työntekijää sai potkut. Yhtiön kerrottiin jatkavan alle kymmenen hengen tiimillä pelilisensointia. Historian oikkuna Lucasfilmin peliosasto onkin siis nykyisin suunnilleen samankokoinen kuin se oli vuonna 1984. Se on muun muassa Rovion *Angry Birds Star Wars II* -lisenssin takana, mutta on sittemmin ryhtynyt lisensoimaan myös yhtiön varsinaisia peliomistuksia, esimerkkeinä *Grim Fandango Remastered* ja *Day of the Tentacle Remastered*.

Miten olisi seuraavaksi *Rescue on Fractalus! Remastered*?

Luke Arnoldin Fractalus löytyy osoitteesta: lsdwa.com/projects/fractalus. Alkuperäinen Rescue on Fractalus! Internet Archive -verkkokirjastossa: archive.org/search.php?query=Rescue on Fractalus.

Lisää yleisemmin fraktaaleista ja maisemakohinasta mm. Skrolleissa 2013.1 (s. 34), 2015.1 (s. 46) ja 2017.1 (s. 72), pdf-versiot: skrolli.fi/numerot.

JOKin PELIPALSTA

BATTLETECH

Lauta- ja roolipelistä ikivihreisiin videopeleihin

Mahtavia sotakoneita. Lasereita, ohjusryöppyjä ja ionikanuunoita. Sotureita, seikkailuja ja tarinoita.

Teksti: Jukka O. Kauppinen
Kuvat: JOK, Mobygames, Piranha Games, Harebrained Schemes

Vuonna 1984 julkaistu *BattleTech*-figuuripeli on kutkuttanut sotaisampien ja seikkailullisempien pelien ystäviä monella tapaa. Siitä on tehty oiva *MechWarrior*-roolipeli, keräilykorttipeli, kolikko- ja virtuaalitodellisuuspelejä, animaatioita ja kasoittain kirjoja. Alkuperäinen figustrategia on saanut vuosikymmenten mittaan myös monia jatkopainoksia ja laajennuksia. Monille *BattleTech* on silti tutuin videopeleistä.

BattleTechin asetelma on monin tavoin kiinnostava. 3100-luvun galaksi on jakaantunut keskenään jatkuvasti kiistelevien mahtivaltioiden kesken. Rikkaaksi kasvaneessa tarinaversumissa on useita päähenkilöitäkin, mutta usein pääosaa esittävät palkkasoturiyksiköt ja näiden sotakoneet, 20–100 tonnia painavat valtavat BattleMechit. Tarinoista puhkuu monesti lähes sotakonepornoa, mutta parhaimmissa kirjoissa pureudutaan myös suurvaltapolitiikkaan, ihmisyyteen ja henkilökohtaisiin tarinoihin.

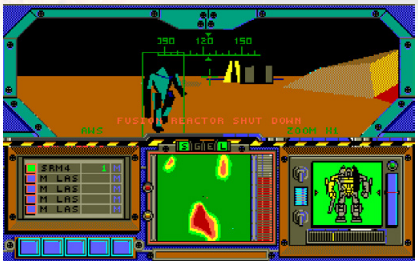
Roolipelistettyä mechsotaa

BattleTech – Crescent Hawk's Inception
Westwood/Infocom, 1988
PC, Amiga, Apple II, Atari ST, Commodore 64

Westwood Studios ehti tehdä monia oivallisia ja jopa nerokkaita pelejä ja pelisarjoja, ennen kuin firma kompastui *Command & Conqueriin*. Peli kertoi oivallisen tarinan nuoresta MechWarrior-oppilaasta, joka joutuu yllättäen tositoimiin. Toisin kuin monissa myöhemmissä *BattleTech*-peleissä, *Crescent Hawkissa* painopiste on roolipelaamisella ja seikkailulla. Sotimistakin on, mutta se on kuvattu samalla ylhäältä kuvatulla mekaniikalla kuin muukin peli: vähän kömpelösti ja hitaasti.

Käsikirjoitetuksi roolipeliksi *Hawkin* tarinaan on upotettu yllättävän paljon satunnaisuuksia ja käänteitä, joiden ansiosta pelikokemus saattaa olla eri pelaajilla hyvinkin erilainen. *BattleTechin* ensimmäisenä videopelimaistiaisena elämys koukuttikin monet perusteellisesti pelisaagaan.

3D-mechien lumo



MechWarrior
Dynamix/Activision, 1989
PC, Sharp X68000, PC-98

Muun muassa simuaattoreista ja *Tribes*-sarjasta muistettu Dynamix tarttui *BattleTechiin* omintakeisella näkökulmalla. Vaikka *MechWarrior* muistetaan ennen kaikkea ensimmäisenä 3D-grafiikkaa ja FPS-näkökulmaa viljelleenä mechistelynä, niin pelissä on myös aimo annos seikkailua ja resurssienhallintaa. Pelaaja astuu syvälle *BattleTech*-versumin ytimeen palkkasoturiosaston komentajana, jonka täytyy värvätä sotureita ja koettaa suoriutua asiakkaidensa sotatilauksista voitollisesti.

Ensimmäinen *MechWarrior* on nykysilmin ja -korvin karu. AdLib-äänet ja rasteripolygonit eivät hivele, mutta hauskaa sotiminen on. Todelliseen kukoistukseen sarja nousi kakkososassa (1995), kun 3D- ja äänikortit olivat kehittyneet monen jättiloikan verran. Pelin taktinen kulma ja taistelujen tunnelma olivat huipussaan, joskin roolipelimäisyys jäi pois. Sarjan myöhemmissä lisäreissä ja jatkoosissa menetettiin osa kakkosen tunnelmasta, ja vaisun nelosen jälkeen pääsarja päätyikin pitkälle tauolle.

Shareware-BattleTech oli liian hyvä

MechForce / MechCombat / BattleForce
Ralph. H. Reed, 198x
Amiga

Kaikkien aikojen paras *BattleTech*-versiointi on se vähiten tunnettu. Yhdysvaltalainen **Ralph H. Reed** ohjelmoi aivan oman *BattleTechinsä* 1980-luvun lopulla. BattleMech-osastojen väliseen taisteluun keskittyneessä strategiapelissä ei ollut tarinaa, mutta sitäkin syvällisemmät hahmoja 'Mech-editorit – ja uskomattoman hyväksi viilattu pelattavuus. Pelijärjestelmä huomioi alkuperäisen lautapelin säännöt ja 'Mech-sodankulun logiikan tarkemmin kuin mikään muu *BattleTech*-peli koskaan, kaiken lisäksi erittäin hyvin suunnitellussa ja pelattavassa paketissa.

Sharewarena levitetty peli koki vuosien mittaan muutaman mittavan muodonmuutoksen, etenkin sen jälkeen, kun pelisarjan oikeudet omistava FASA käveli Reedin päälle ja käski tämän lopettaa puuhastelunsa. Hänen *BattleTechinsä* kun oli parempi kuin ne viralliset. Reed ei enää levittänyt peliä julkisesti, mutta rekisteröityneet pelaajat saivat vielä muutaman pidemmälle kehitetyn version yksityisesti.

Reaaliaikanaksua vihollisen selustassa

MechCommander
FASA Interactive / MicroProse, 1998
PC

BT-versumin parempaan päähän sijoittuva *MechCommander* muunsi jatko-osineen saagan huikeaksi reaaliaikastrategiaksi, jossa riitti ainutlaatuista kuvakulmaa, resurssienhallintaa ja kustomoitavuutta. Klaanisotien aikaan sijoittuvassa pelissä johdettiin enimmillään 12 BattleMechin osastoa tekniikaltaan ja lukumäärältään ylivoimaista vihollista vastaan. Haastetta siis riittää mutta niin myös palkitsevuutta.

Yksikön, 'Mechien, talouden ja sotureiden hallinnointi on jo itsessään täysimittainen ja syvä strategiapeli. Lisäksi reaaliaikainen sodinta eri aselajien yhdistelmineen piti otteessaan niin hyvin, että peli viihtyy edelleen fanien käsissä. Harmillisesti sitä ei ole julkaistu uudelleen esimerkiksi GOG.comin kautta, mutta sen voi nähtävästi ladata nykyään ilmaiseksi.

Paras BattleTech, jota ei ollut

Multiplayer BattleTech: 3025
Kesmai, 2001
PC

Jenkkiläinen Kesmai oli 1990-luvulla yksi nettimoninpelien pioneereista ja kovimmista tekijöistä. Firman 2001 betaan julkaistu, 2001 tapettu *BT: 3025* oli – tai olisi ollut – yksi firman parhaimmista peleistä ikinä, jos sille olisi annettu mahdollisuus. Mutta valitettavasti Kesmain ostanut EA teki tylysti ja pisti studion kiinni dot.com-kuplan puhjettua.

Ja sekös rassasi, sillä *3025* oli harvinaisen toimiva nettipelattava MMO-sotapeli. Syytä olikin, olihan se jo Kesmain kolmas *BT*-moninpeli. Neljän pelaajan tiimit kahakoivat keskenään erittäin pikkutarkasti tehdyssä simulaattorissa, ja tulokset heijastuivat suurempaan metapeliin. Simulaattorina *3025* pieksi jopa MechWarriorit, sillä se huomioi alkuperäispelin säännöt tarkemmin ja realistisemmin. Tulos oli erittäin toimiva nettielämys. Jälkikäteen puntaroituna *3025* oli yksi parhaista kaverien kanssa pelattavista nettipeleistä ikinä.

3025:n jälkeen 'Mech-nettirintamalla oli pitkään hiljaista, mutta tänään sen jalanjäljillä marssii Piranha Gamesin ilmainen *MechWarrior Online*. Sen taival on tosin ollut kuoppainen. Nykyisellään pelin suurin ongelma on ilmeisesti BT-aikajatkumon turhan vauhdikas juoksuttaminen eteenpäin.

Hyvää luvassa!

BattleTech on ollut aika lailla sivuraiteella 2000-luvun alun jälkeen. Tänä vuonna kuitenkin rytisee. Piranha Games työstää *MechWarrior 5: Mercenaries* -peliä (joulukuu 2018) ja koko pelisarjan alkuperäinen luoja, **Jordan Weisman**, palaa kentälle. Herran Harebrained Schemes -studio näet tekee *BattleTech*-nimellä marssivaa vuoropohjaista strategiapeliä, joka ainakin allekirjoittaneen alustavan testailun perusteella tuntuu parhaimmalta *BattleTechiltä* sitten *MechForcen*.

Harebrainedin pelistä näkee, että nyt asialla on mestari itse. Se huomioi hienosti asioita, joita useimmissa BT-teoksissa ei ole viitsitty toteuttaa, ja rakentaa 'Mech-sodan uniikilla pienyksiköiden perspektiivillä. Tarinakaan ei kiepu suurvaltapolitiikan ympärillä, vaan avaa pelaajille galaksin vähemmän tunnettuja alueita.

Joten kiillottakaa neurokypäränne. Kohta kelpaa kääntää virta-avaimesta.

Ei näin! Espanjassa surisee

Teksti: Mikko Heinonen
Kuvat: Mikko Heinonen, Manu Pärssinen, PlayStation Europe, Wikimedia Commons/ADwarf

E-sports-urani huipentui MM-hopeaan, jota tuskin koskaan joudun puolustamaan.

Ei näin! on keskittynyt Skrollin viiden vuosikerran (sekä nollanumeron) ajan isoihin epäonnistumisiin, joissa peli floppaa tai kokonainen laitevalmistaja menee konkurssiin. Tällä kertaa haluan kuitenkin kertoa hieman henkilökohtaisemman tarinan siitä, kuinka moni asia menee pieleen.

Olen kirjoittanut tapauksesta aiemman jutun, joka löytyy yhä verkosta. Se on sävyltään positiivinen, ja jätän tarkoituksella asioita kertomatta, koska olin lopulta kuitenkin mielissäni siitä, että sain mahdollisuuden olla mukana erikoisessa tapahtumasarjassa, enkä halunnut pahoittaa niiden mieliä, jotka auttoivat minua siinä. Nyt, kun kaikesta on kulunut jo liki 10 vuotta, ovat useimmat asiaan liittyvät jo jossain muualla töissä. Tässä siis kokonaisuudessaan sensuroimaton kertomus ensimmäisistä ja viimeisistä Buzzin MM-kisoista.

Oho, osaan tämän

Silloin kun en larppaa journalistia, tienaan elantoni kielenkääntäjänä. Ammatin pääsyvaatimus on kielitaidon lisäksi se, että tietää kaikenlaista ja osaa tarvittaessa ottaa asioista selvää. Yhdessä pohjattoman uteliaan luonteeni kanssa tämä jotenkin johti siihen, että huomasin jossain vaiheessa olevani aika hyvä tietokilpailuissa.

Buzz! oli nimenomaan konsolille siirretty tietovisa. Ideassa sinällään ei ollut mitään uutta, sillä näitähän oli nähty jo 80-luvulta alkaen, mutta toteutus oli pirteä: pelin mukana tuli erityiset ohjaimet, joilla paitsi napattiin vastausvuoro myös valittiin vaihtoehto. Lisäksi peli oli käännetty suomeksi siedettävän hyvin. Sarja alkoi PlayStation 2:lla ja oli PS3:lle siirryttyään saanut kylkeensä myös erityisen verkkopelin. Sitä markkinoidakseen Sony päätti järjestää MM-kisat.

Karsinta käytiin maatasolla: jokaisen osallistujamaan 32 parasta kutsuttiin loppukilpailuun, josta voittaja pääsisi edustamaan maataan. Tämä asetti itselleni ensimmäisen ongelman: olin nimittäin saanut pelistä pressikappaleen ja siten minun oli tarkoitus osallistua vain ”lehdistösarjaan”. Kun Manun (juuri se sama Skrollin taittaja) yllytyksestä kokeilin peliä ensimmäisen kerran, sijoitukseni Suomen listalla oli viidestoista. Innostuin tästä sen verran, että hieroin itseni listalla ensimmäiseksi, kunnes juuri ennen finaalia minut ohitti nimimerkki **Tissiposki**. Tällä nyt ei tosin ollut merkitystä, sillä 32 parasta sai kutsun – enkä edes tiennyt, pääsenkö pelaamaan muussa kuin lehdistöturnauksessa.

Asioita tapahtuu

Saavuttuani Helsinkiin sain kuulla järjestäjiltä, että voin aivan hyvin pelata pääsarjassa. Elettiin jo aikaa, jolloin pelit päivittyivät verkosta, joten saamani lehdistöversio oli täysin samanarvoinen kuin kaupasta ostettu.

Olin toki pelannut itseni listaykköseksi ja oppinut siinä monia pelin kysymyksistä, mutta muuten en voi sanoa harjoitelleeni. Siitä huolimatta voitin ensin oman karsintaeräni, sitten välierän ja lopulta huomasin pelaavani koko turnauksen voitosta. Kun **Sampo Marjomaa** sitten kätteli minut Suomen Buzz-mestariksi, toivotti hyvää matkaa MM-kisoihin ja finaalivastustaja kertoi tienneensä koko ajan häviävänsä, tunnelma oli aika epätodellinen: en ollut käsittänyt olleeni ennakkosuosikki. Tietysti olin 33-vuotiaana hieman vanhempi kuin monet pelaajista, mutta en nyt sentään vuosikymmeniä.

Tähän treenausmusiikkia

Kun sain kuulla lähteväni Espanjan Valenciaan pelaamaan maailmanmestaruudesta, päätin ottaa asian tosissani.

Pelasin peliä käytännössä joka päivä ja kävin jopa ostamassa langalliset Buzz-ohjaimet, kun kävi ilmi, että langattomien ohjainten viiveen vuoksi menetin jopa 100–150 pistettä kierrosta kohti. Kirjoitin vaikeiden kysymysten vastauksia vihkooni ja hioin vastaustekniikkaani. Koska osasin odottaa tuoreita musiikkikysymyksiä, pyöritin Spotifyssä soittolistaa radiohiteistä.

Tästä huolimatta omat pisteeni eivät olisi riittäneet monenkaan Euroopan maan listan kärkeen. Koska osasin pelin käytännössä ulkoa enkä voinut enää nopeuttaa omia vastauksiani, arvelin tämän johtuvan kielieroista. Välillä kokeilin pelata peliä toisella kielellä ja huomasin, että osa kysymyksistä on samoja, vaikka suuri osa olikin lokalisoitu varsin ansiokkaasti.

Elokuun lopussa oli sitten aika astua kisakoneeseen. Sain ottaa mukaani yhden seuralaisen, joten tuolloin 3-vuotias tytär jäi vanhempien luokse, ja matkustimme vaimoni kanssa. Koska lento lähti varhain aamulla, sain puhuttua meille edellisen yön lentokenttähotellissa Helsinki-Vantaalla. Se sijaitsi tuolloin konkreettisesti lähtöportin vieressä, enkä tunnetusti herkkäunisena nukkunut kerosiinin katkussa ja jettien jylinässä montaakaan tuntia. Univaje pohjustikin hyvin seuraavan päivän absurdiutta.

Espanja pikakelauksella

Aamulla marssimme lentoyhtiön tiskille tekemään lähtöselvitystä. Se onnistui vain puoliksi: pääsisimme Finnairin lomalennolla Palma de Mallorcalle. Sieltä luvattuun jatkoyhteyteen emme kuitenkaan saaneet lippuja. Tästä olisi pitänyt olla huolissaan, mutta arvelimme kaiken järjestyvän – olimme kuitenkin matkalla jonnekin, vaikkakin nousuhumalaisten rantalomaturistien seassa. Mukaan lähti jopa Iltalehden toimittaja.

Totuus alkoi paljastua Palma de Mallorcalla, kun havaitsimme, että koska lähtöselvitystä ei ole tehty, meidän pitää poistua saapuvien aulan kautta ja kulkea sitten check-in-tiskille uudestaan. Ja sen jälkeen uudestaan turvatarkastukseen. Koneen lähtöön oli aikaa alle tunti, joten huumori oli tiukalla.

Tiskille päästyäni sain sitten kuulla syyn sille, miksi lähtöselvitys ei Suomessa sujunut: Valencian kone on ylibuukattu, emmekä pääse siihen. Pidän käsittämättömänä, että tästä ei mainittu meille lähtökentällä sanallakaan. Espanjalainen asiakaspalvelija latoi ilmeenkään värähtämättä tiskiin korvauksena 500 euroa, siis 250 euroa per matkustaja, mutta se ei paljon lohduttanut, sillä MM-kisat pelattaisiin samana iltana ja olimme satojen kilometrien päässä tapahtumapaikalta. Kaivoin puhelimen esiin ja soitin järjestäjille. Sain kuulla, että Ibizan kautta on vielä yksi lento, jolla voisimme ehtiä. Siispä lippujen vaihto ja Ibizan-koneeseen.

Seuraava tunti kuluikin sitten Mallorcalla kiitotien päässä, koska Ibizan lentokenttä oli suljettu, eikä kone saanut lähtölupaa. Lopulta lupa kuitenkin heltisi ja olimme taas matkalla kohti Ibizaa, ja lentoemäntä kertoi Valencian-koneen odottavan meitä kentällä.

Laskeuduttuamme Ibizalle seikkailussa siirryttiin seuraavalle vaikeustasolle. Matkustajat ohjattiin koneesta lentokenttäbussiin, jolla oli tarkoitus palata terminaaliin. Yhtäkkiä bussissa alkoi espanjankielinen kuulutus, josta kuin jumalaisella intuitiolla poimin sanan ”Valencia”. Samassa hetkessä huomasin, että bussin vieressä on toinen lentokone ja sen jonossa seisoo nainen, jonka tiesin vaihtaneen omat lippunsa tiskillä meidän edellämme. Saimme hypättyä pois bussista juuri ennen kuin se lähti liikkeelle ja pääsimme kuin pääsimmekin lopulta oikeaan koneeseen.

Valenciassa meitä oli vastassa Mercedes-Benz E-Klasse ja kuljettaja, jonka hyperaktiivisesta käytöksestä ja tavasta ajaa ylinopeutta kasvot takapenkille käännettynä tuli mieleen jokin toinen E. Autossa näytti mittarin mukaan olevan bensaa jäljellä 30 kilometriksi, mutta kiihtyvyyden siunatun jumalattaren suosiollisella avustuksella pääsimme hotellille, jossa ehdimme hädin tuskin vaihtaa vaatteet ennen kisoihin lähtöä.

Ei näin: Jos järjestät MM-kisat, älä varaa kilpailijoiden lentoja kisapäivälle minuuttiaikataululla. Kukaan ei halua joutua pelaamaan reissussa rähjääntyneenä, väsyneenä ja nälkäisenä. Oli useamman kerran aivan hiuskarvan varassa, että ylipäätään pääsimme perille, ja vähänkin itseäni kokemattomampi olisi varmasti jäänyt koneesta.

Tomaattiaattoilta

Valencia oli valittu kisapaikaksi La Tomatina -festivaalin vuoksi. Tässä järjen riemuvoitossa 40 000 ihmistä kerääntyy heittelemään toisiaan tomaateilla tunnin ajaksi – kaikkiaan ilmoille viskotaan noin 100 tonnia vihanneksia. MM-kisojen voittaja saisi kunnian heittää ensimmäisen tomaatin. Meille kerrottiin matkalla, että kisat pelattaisiin tomaattifestivaalin ”aattona” ja sen vuoksi paikalla olisi tuhansia ihmisiä juhlimassa. Ehdin jo mietiskellä, miltä tuntuu pelata valtavan yleisön edessä.

Kun bussi sitten kaartoi kisapaikaksi valitun hiekkakentän reunaan, en nähnyt siellä juuri ketään muita ihmi-

Tänään Hauskoissa kotivideoissa: Absurdin tapahtumaketjun alku.

siä, ja todennäköisesti kaikki paikalla olleet liittyivät jotenkin turnaukseen. Eikä ihme, sillä järjestäjät olivat suuressa viisaudessaan päättäneet periä alueelle tulevilta pääsymaksun. Voin kuvitella, että Etelä-Euroopan kesän ollessa kauneimmillaan keksisin muutakin tekemistä kuin maksaa siitä, että menisin katsomaan, kun joukko ulkomaalaisia pelaa Pleikkarilla.

Se mikä ihmisissä hävittiin, voitettiin pölyssä. Aurinko oli paahtanut vanhan betonitehtaan vieressä sijainneelle hiekkakentälle koko päivän ja kuivattanut valtavan määrän hienojakoista pölyä, jota (harvojen) ihmisten kävely ja tuulenpuuskat nostivat ilmaan. Tuona iltana otetuissa valokuvissa näkyi runsaasti parapsykologian harrastajien tavoittelemia niin sanottuja enkelipalloja, minkä lisäksi pölyä kertyi kenkiin, vaatteisiin ja jopa hiuksiin. Maistoin sen suussani monta päivää tuon illan jälkeen.

Ei näin: Sisäänpääsymaksu. Visailupeliturnaukseen. Oikeasti?

Taisto alkaa

Pian pölykentälle saapumisen jälkeen sain kuulla, että loppukilpailun karsinnat pelattaisiin vielä julkaisemattomalla *Buzz! Visailun maailma* -pelillä. Kukaan pelaajista ei ollut nähnyt sitä aiemmin, joten pidin tätä erinomaisen reiluna ratkaisuna. En luonnollisestikaan olettanut, että vanhaa Buzzia pelattaisiin, sillä en varmasti ollut ainoa sen kysymykset ulkoa opetellut.

Sen sijaan huomattavasti huonommalta kuulosti se, että pelaajat oli jaettu neljän hengen eriin, jotka pelaisivat lavalla, kun muut odottaisivat katsomossa – ja että karsintaerien kysymykset heijastettaisiin isoille näytöille pelin aikana!

Menin suoraan kilpailupäällikkönä toimineen PR-ihmisen puheille ja sanoin, että tässä on mielestäni massiivinen ongelma. Jos muut saavat katsella toisten peliä, on aivan selvä, että myöhemmissä erissä pelaavat saavat etua kysymysten näkemisestä. Vastaus oli hämmentävä: tästä ei ole mitään haittaa, koska pelaajat puhuvat eri kieliä. Selitin, että aiemman Buzz-version kokemusten perusteella jonkinlainen osa kysymyksistä on suoraan käännettyjä, ja että esimerkiksi itse osaan liki äidinkielisen tasolla englantia, minkä lisäksi ymmärrän ruotsia, ranskaa ja ranskan

Minä 20 kg sitten, Armin ja Ian.

pohjalta jonkin verran italiaa ja espanjaa, korkkarisaksasta puhumattakaan. Ottaen huomioon, millaista porukkaa tietokilpailut houkuttelevat, en uskonut olevani lainkaan ainoa kieli-ihminen. Tuntui vaikealta edes käsittää, ettei asiaa ollut ajateltu.

Tuomaristo kokoontui asiasta ja teki päätöksen, että katsomonäytöt suljettaisiin karsinnan ajaksi. Tämä oli voitto rehdille pelille, mutta tappoi varmasti lopunkin mielenkiinnon kisan seuraamiseen vähälukuisen yleisön parissa. Sori siitä.

Päädyin itse ensimmäiseen karsintaerään, jossa sijoituin kolmanneksi. Kahdeksan parasta kaikkiaan neljästä erästä oli pääsemässä jatkoon, joten arvelin siirtyväni katsomon puolelle. Sattuipa kuitenkin niin, että oma erässäni pelasivat lopulta sijoille 1, 2 ja 3 sijoittuneet. Kukaan muista eristä ei pystynyt ohittamaan edes minun pisteitäni, joten olin kirkkaasti jatkossa.

Ei näin: Kun järjestetään peliturnausta, voi kannattaa konsultoida pelaajia. Jo ennen kuin tilaa paikalle useita jättinäyttöjä, joita ei sitten voi käyttää.

Distraction!

Semifinaali pelattiin samalla tavoin videopelillä kuin alkuerätkin, mutta nyt saimme yllemme valkoiset haalarit. Kierroksen häviäjää heitettäisiin ”kakulla”, eli hätäisesti lautaselle sutaistulla marjavaahdolla. Kahdesta kakutuksesta putoaisi pois.

Omaan suoritukseeni tuli mukavasti painetta, kun pelaamani suomenkielinen versio alkoi sekoilla heti ensimmäisellä kierroksella. Kaiuttimista kuului luettuna eri kysymys kuin ruudulla näkyi, enkä aidosti tiennyt, kumpaan pitäisi vastata. Valitsin luonnollisesti väärin ja pisteet menivät nollille. Tämä hämmensi niin, että pelasin koko illan huonoimman tulokseni ja jouduin kakutettavaksi heti aluksi. Ilmoitin asiasta tuomaristolle ja sain vastaukseksi käsien levittelyä, ”peli ei ole ihan valmis”. Jatkossa onneksi pelasin omalla tasollani ja tiesin, miten vastata bugin sattuessa, joten selvitin tieni finaaliin kakutuksesta huolimatta.

Ei näin: Pelin MM-finaali ei ole betatestauspaikka, eikä ole hirveän reilua rangaista pelaajaa siitä, että peli ei toimi.

Yllättävä tasapeli

Finaalissa oli lopulta neljä maata: Iso-Britannia, Itävalta, Suomi ja Sveitsi. Pelityyppi muuttui, sillä maakohtaisten kysymysten sijaan siirryttiin eri-

tyiselle DVD-levylle koottuihin ”kansainvälisiin” kysymyksiin, joita pelin juontajaksi puettu illan isäntä spiikkasi. Ensimmäinen osio käsitteli musiikkikappaleiden tunnistusta, ja tiesin jo etukäteen, että Isoa-Britanniaa edustanut skotti on siinä aivan pitelemätön. Hän dominoikin kisaa siinä määrin, että muiden tehtäväksi jäi napata edes yksi oikea vastaus, jotta välttää putoamisen. Itse onnistuin siinä, itävaltalainen kilpailija ei, joten jäljelle jäi kolme pelaajaa.

Seuraava laji oli päättelytehtävä, jossa piti valita, mikä kuvista ei kuulu joukkoon. Tämä tuntui sopivan omille aivoilleni hyvin ja sain kuusi pistettä 15:stä, muut pelaajat viisi kumpikin. Tie-breakerissa sveitsiläinen pelaaja järjesti yllätyksen ja tiputti skotin, jota itse ennustin alkuillasta koko kisan voittajaksi.

Viimeisessä kisaosiossa piti tunnistaa kuvia, jotka ilmestyivät sumennettuina näytölle. Aloitin vahvasti ja pääsin hyvään johtoon, mutta sveitsiläinen tuli aivan kannoille. Lopulta ennen viimeistä kuvaa pistetilanne oli 7–7. Muistan selvästi tätä seuranneen tilanteen. Juontaja jäi jaarittelemaan jotain edellisestä kuvasta, kun ohjaamosta laitettiin jo seuraavaa pyörimään. Huomasin tämän ja tunnistin kuvan esittävän mustavalkoista kissaa. Painoin summeria ja vastasin oikein.

Sen sijaan, että kisa olisi päättynyt tähän, tuomaristo vetäytyi ”laskemaan pisteitä”. Tiesin, että jos viimeinen kuva laskettaisiin, olisin voittanut. Varsin pitkän hetken kuluttua taululle ilmestyi pistetilanteeksi 7–7, mikä oli varsin mielenkiintoinen lopputulos 15 kuvan jälkeen. Luvassa olisi taas tie-breaker. En saanut koskaan selitystä sille, miksi viimeinen kuva hylättiin: olin toiminut sääntöjen mukaisesti, vaikka kuvaa ei juonnettukaan sisään.

Ei näin: Kun pelataan kymmenien tuhansien palkinnosta, älä sählää järjestelyjä. Jos oltaisiin oltu Amerikassa ja palkinto olisi ollut isompi (ks. alla), tästä olisi voitu käydä oikeutta.

Ei edes harmita

Hyvä puoli tuomareiden pitkässä tuumaustauossa oli se, että ehdin jutella Sveitsin edustajan kanssa. Olin nimittäin mielenkiinnosta selvittänyt, millainen veroseuraamus minua Suomessa odottaisi, jos sattuisin voittamaan pääpalkintona olleen Mini-merkkisen henkilöauton. Koska kyseessä oli taitopeli eikä arvonta, auto olisi veronalaista ansiotuloa. Näin joutuisin maksamaan noin kolmanneksen sen hinnasta veroa eli käytännössä myymään auton saman tien niiden maksamiseksi. Toki olisin yhä jäänyt tuhansia euroja plussalle, mutta loppuvuoden veroprosentti palkkatöistä olisi alkanut nelosella (tai sitten vuoden päästä olisi ollut odotettavissa kaikkien aikojen mätkyt). Samasta syystä esimerkiksi urheilijat ovat jättäneet turnausten voittoautoja lunastamatta.

Kysyin sveitsiläiseltä suoraan, tiesikö hän palkinnon verokohtelusta omassa kotimaassaan. Kyllä tiesi – se olisi hänelle verovapaa. Seuraava kysymykseni oli, millainen auto hänellä nyt on. Olin itse ostanut noin vuotta aiemmin miltei uuden mukavan perheauton, joten minulla ei ollut senkään puolesta mitään intohimoja melko perusmallisen Minin suhteen. Armin kertoi ajavansa 15-vuotiaalla Audilla.

En aio valehdella, että hävisin tahallani tie-breakerissa. Kolmen kuvan sarja ratkesi siihen, että tilanteessa 1–1 Armin tunnisti Will Smithin sekunnin murto-osan ennen minua. Mutta käytyäni tuomaritauon aikana keskustelun hänen kanssaan, minusta alkoi tuntua, ettei minua harmita hävitä kisaa juuri tälle vastustajalle. Hän on ansainnut palkinnon siinä kuin minäkin, saa pitää sen kokonaan ja se parantaa hänen elintasoaan välittömästi. Itse joutuisin kikkailemaan yhtä jos toista, minkä jälkeen nettohyötyni auton voitosta olisi parin kuukausipalkan verran. Rahaa toki sekin, mutta silti.

Joten vaikka hopea hävitään, koin myös voittaneeni. Yli 20-tuntiseksi venynyt päivä oli ollut sellaista vuoristorataa, että ihmettelin, etten vielä ollut täysin tiltissä, vaan olin pystynyt pelaamaan itseni MM-hopeamitalille. Lisäksi vältin etukäteen jännittämäni La Tomatina -festivaalin, ja menimme sen sijaan vaimoni kanssa seuraavana päivänä junalla Valenciaan tuhlaamaan lentoyhtiön antamat 500 euroa.

Minua ei kiukuttanut edes se, että palkinnot oli todellakin porrastettu jyrkästi: voittaja sai auton, kakkonen nenäkarvatrimmerin, kolmonen leivänpaahtimen. Ratkaisu mukaili pelin hassuja palkintoja, mutta vähemmän zen-tilassa tätäkin olisi voinut pitää hieman loukkaavana. Seuraavana päivänä saimme toki kuulla, että finalistien palkintopakettiin kuuluu myös julkaisuaan odottava PSP Go, levyasematon muunnos Sonyn käsikonsolista. En muuten tiedä ketään, joka olisi maksanut sellaisesta täyden jälleenmyyntihinnan.

Ei näin: PSP Go. Fail.

Nero 2000 – älypelien legenda

Digiarkeologit sukelsivat klassisen tietovisan saloihin

Tietovisailu Nero 2000 on pitänyt paikkansa suomipelien klassikkona jo 30 vuotta.

Teksti: Aki Sivula
Kuvat: Toni Kortelahti, Aki Sivula, Mika Hyvönen

IRC on jännä paikka, sillä sen kanavilla käydyistä keskusteluista käynnistyy joskus merkityksellisiä asioita. Heinäkuussa 2016 Kasettilamerit-entisöintiryhmän (ks. Skrollit 2017.4, 2013.1) IRC-kanavalla keskusteltiin 1980-luvun suositusta *Nero 2000* -tietovisapelistä, joka oli saatavilla C64-, PC- ja MSX-laitteille. Pohdimme kuinka hienoa olisi tehdä pelin C64-versioon uusia kysymyksiä. Lisäksi peliin olisi hienoa toteuttaa muutamia teknisiä parannuksia.

Olimme julkaisseet aiemmin *Arthur The Little Knight* -pelin tekijänsä **Pasi Hytösen** luvalla, joten tässä olisi ryhmän toinen julkaisu. *Neron* tekijänoikeudet ovat kuitenkin edelleen voimissaan, joten ensimmäiseksi pitäisi löytää sen tekijät ja selvittää, onko pelin julkaissutta firmaa enää olemassa, tai kuka pelin oikeudet ylipäätään omistaa. Saataisiinko heiltä lupa uuden version toteuttamiseen? Voiko uutta versiota ylipäätään julkaista?

Juho Kuorikosken ansiokkaasta *Sinivalkoisesta pelikirjasta* ei ollut tällä kertaa apua salapoliisityössä, joten lähdin selvittämään asiaa. Pieni googlailu, sähköpostiosoitteiden kokeilu ja Facebookissa haku tuotti tulosta melko pian. Lopulta olin puhelimessa taannoin lopetetun Bio-Syntax Methodin omistajan kanssa. Lopputuloksena sain tallentaa pelistä alkuperäiset Commodore 64- ja PC-versiot talteen Software Preservation Societyn arkistoihin, haastatella pelin kehittäjätiimiä sekä toimittaa materiaalia Suomen Pelimuseoon.

Nero 2000 – unohtumatonta visailua

Kulttimaineeseen noussut *Nero 2000* julkaistiin 30 vuotta sitten, ja siitä tuli ensimmäinen kaupallisesti menestynyt suomalainen tietovisailupeli. Peli muistuttaa jossain määrin *Trivial Pursuitia*, mutta on kuitenkin täysin itsenäinen tuote. Peliä pelataan joko kahdestaan tai tietokonetta vastaan.

Tavoite on voittaa toinen pelaaja sytytettyjen lamppujen tai pisteiden avulla. Peli päättyy, kun toisen pelaajan pisteet loppuvat tai pelaaja on saanut kaikki lamput syttymään. Kysymyksiä on 500 neljästä eri kategoriasta, ja vastaukset kirjoitetaan näppäimistöllä monivalintavastaamisen sijaan.

Neron legendaarisin ominaisuus on vastausten ehdottomuus: pelaaja

joutuu arvuuttelemaan oikeaa synonyymi-, sana- tai sijamuotoa tai jopa kirjoitusasua. Usein oikea vastaus menee väärin jonkin pikkuseikan vuoksi, mikä saattaa aiheuttaa turhautumista. Peli on kuitenkin aikansa kuva ja erinomainen tuote, joten se sai erittäin hienosta toteutuksesta kehuja **Max Hammarbergiltä** *Mikrobitin* 11/87 koko sivun artikkelissa.

Suomen pelimuseon mukaan ”*Nero 2000*:n klassikkoasema onkin osittain ansaittu sen aiheuttamien harmaiden hiusten vuoksi”.

Neroa myytiin kasetti- ja diskettiversiona Infon kirjakaupoissa, Akateemisessa kirjakaupassa, Mustassa pörssissä, Expertillä ja käytännössä kaikissa kodinkonekaupoissa. Suurimmaksi osaksi myynnissä oli punakantisia versioita, mutta Akateemiselle kirjakaupalle tehtiin oma huomattavasti harvinaisempi vihreäkantinen versio. Infossa Commodore 64 -levyversion hinta oli 148 markkaa ja kasettiversion 128 markkaa. Tarkkaa myytyjen kappaleiden määrää ei enää ollut muistissa, mutta arvio liikkui tuhansissa kappaleissa. Ensimmäisenä kauppoihin saatiin hieno Commodore 64 -versio vuonna 1987 ja hieman myöhemmin vaatimattomammat MSX- sekä PC-versiot (1988). Versioiden ulkoasu poikkeaa toisistaan melkoisesti.

Nero 2000:n tarina sai alkunsa useasta onnekkaasta sattumasta sekä innokkaasta pioneerihengestä. Bio-Syntax Methodia ei olisi ilman **Taisto Orrea** ja **David Cumberworthia**. Miehet perustivat yrityksen Hyvinkäälle vuonna 1985 tarkoituksenaan tarjota englannin kielen koulutuspalveluja. Kyseessä oli siis kielikoulu tavanomaisten keskustelukerhojen kera.

David oli Ison-Britannian kansalainen, joten hänen vaimonsa **Heidi** oli yrityksen äänetön yhtiömies. Markkinointijohtajana toimi **Kaj Lassander** sekä myyntiedustajana **Seija Hirvinen**. Orre oli yrityksen johtaja ja loi pelin kysymykset, Cumberworth puolestaan oli ideanikkari ja ohjelmoija sekä vastasi kieliasusta. Cumberworth oli toiminut aiemmin Lontoossa bobbyna, paikallisena poliisina, ja sittemmin Saksassa laivaston leivissä. Hän kiinnostui koulutuksesta ja ohjelmoinnista, mutta hänellä ei ollut pedagogista taustaa. Yrityksen ensimmäiset Commodore 64- ja PC-kieliohjelmat olivat tästä johtuen joidenkin pedagogisesti pätevöityneiden

kouluttajien mukaan liiaksi murteeseen painottuneita. Yritys kehitti kuitenkin oman Bio-Syntax Method -nimisen menetelmän, joka jalkautettiin lopulta tuoteperheeksi.

Englannin kielen opiskeluohjelman ohjekirjassa vuodelta 1987 todetaankin seuraavasti: ”Testiemme mukaan ohjelmat sopivat erinomaisiksi apuvälineiksi esim. suggestopedisen opetuksen rinnalle.”

Kuinka Nero 2000 syntyi?

Alkujaan koulutukseen liittyvän koneellisen opetusmateriaalin valmistus tapahtui vain sivutuotteena. Myöhemmin opetus kuitenkin siirtyi lähes kokonaan tietokonevetoiseksi. Koulutuspalvelujen lisäksi ja uudistushaluisen Hyvinkään koulutoimen johtajan suotuisalla myötävaikutuksella Taisto ja David aloittivat kielten opetusohjelmien ja koulutusta tukevien pelien kehittämisen. Tuolloin käynnistettiin kokeilujakso oppimisvaikeuksista kärsiville, ajan termein tarkkailuluokka-

Nerossa kisailtiin neljässä eri kategoriassa.

Osa pelin kysymyksistä on jo vanhentunut, mikä voi kampata junioripelaajan.

Nero 2000:n PC-versio oli C64-versiota vaatimattomampi.

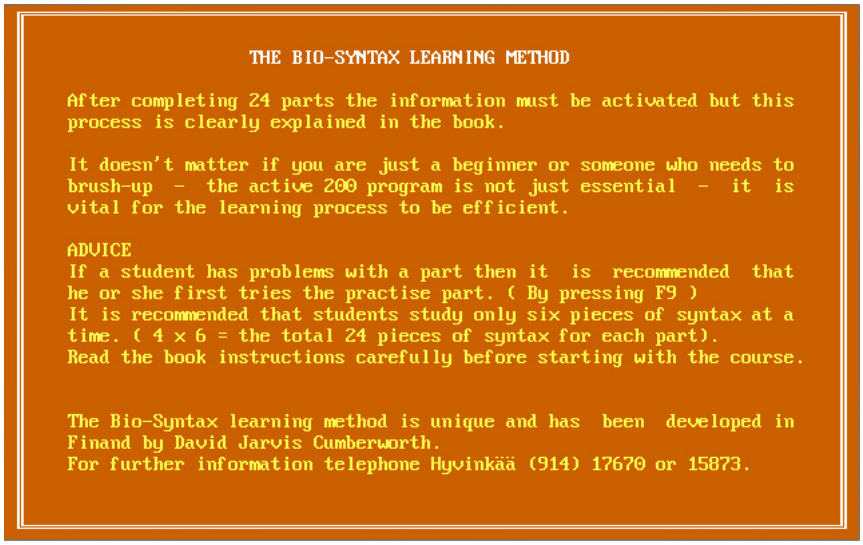
laisille. Kokeilujaksossa erityistä oli tietokoneiden hyödyntäminen kielten opetuksessa. Kokeilu tuotti uskomattoman hienon tuloksen, ja tarkkislaisten kielitaito ylitti pian reippaasti muiden oppilaiden taidot. Kone koettiin tuolloin reiluna ja oikeudenmukaisena opetuksessa, eikä pärstäkerroin päässyt millään tavoin sotkemaan oppilaan ja opettajan suhdetta.

”Tutkimukset osoittivat kieliohjelmiemme sopivan erittäin hyvin oppilaille, joilla on motivaatiovaikeuksia, esimerkiksi erityisluokkalaisille”, tekijät muistelevat.

Nero 2000:n kehityksessä ja muissa kieliohjelmissa oli kuitenkin paljon teknisiä ongelmia, jotka piti ratkaista nopeasti. Kuvaan astui VIC-20:llä aloittanut ja Commodore 64:ään siirtynyt nuori lahjakkuus **Timo Kokkonen**. Työsuhteen alkaminen Bio-Syntax Methodilla tapahtui onnekkaan sattuman, mutta myös lahjakkuuden toteamisen kautta.

Timo oli aloittanut ohjelmointiharrastuksen samaan tapaan kuin moni muukin sen ajan nuori. MikroBittiä tilattiin ensinumerosta alkaen 1984 ja lehdestä kirjoitettiin alkuun listauksia, myöhemmin syntyi jo omia ohjelmia. Eräänä päivänä hyvinkääläinen **Tuukka Uusiheimo** pyysi Timoa tuomaan modeemin Bio-Syntax Methodin toimistolle, jossa painittiin samana päivänä jonkin teknisen ongelman parissa. Timo onnistui ratkaisemaan ongelman, minkä myötä hänet palkattiin kesätöihin Bio-Syntax Methodin ohjelmoijaksi Davidin partneriksi. Näin David pystyi keskittymään enemmän ideointiin.

Timo nostikin ohjelmien teknisen laadun uudelle tasolle. Taiston mukaan ohjelmissa oli aiemmin opetusta häirinneitä bugeja riesaksi saakka, ja



Kieliohjelmat pohjustivat tietovisan syntyä.

Timon mukaan *Nero 2000*:n alkuperäinen koodi tarvitsi niin suuren remontin, että hän kirjoitti pelin Commodore 64 -version käytännössä uusiksi. Peli toteutettiin Commodore 64:n Basicilla sekä Timon itse toteuttamilla Basic-laajennuksilla, jotka olivat käytännössä assemblerilla toteutettua konekieltä.

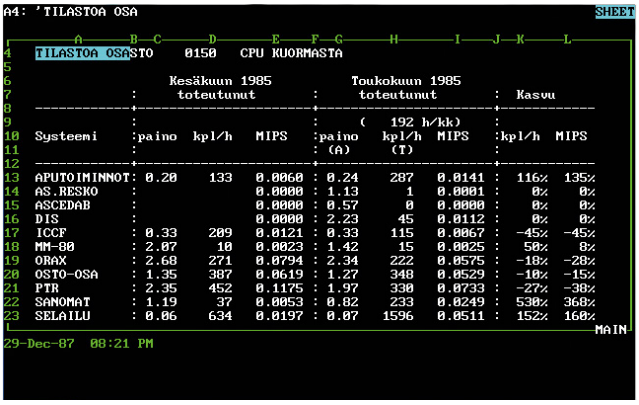
Koodin lisäksi myös grafiikat ovat Timon käsialaa. Taisto valitsi hyväntuuliseksi musiikiksi **Henry Mancinin** *Baby Elephant Walk* -kappaleen, mikä ohjelmoitiin saatujen nuottien perusteella. Teostomaksut hoidettiin asiaankuuluvalla tavalla. Kasettiversioon lisättiin turbolataus sekä levyversioon kopiosuojaus. Pelissä kysymyksiä esittävästä sympaattisesta pöllönpoikasesta tehtiin oma vedos myyntiesitteeseen. Myöhemmin Timo toteutti vielä PC-version, ja vain harvinaisempi MSX-versio jäi täysin Davidin käsialaksi. Kokonaisuudessaan Nero 2000:n toteutukseen meni arviolta puolitoista vuotta.

Yrityksellä oli *Neron* lisäksi erilaisia yrityskäyttöön tarkoitettuja englannin kielen opetusohjelmia PC:lle. Bio-Syntax Methodista tulikin nopeasti suomenkielisten opetusohjelmien ja pelien osalta yksi alan pioneereista 1980-luvulla toimineen Koulun Erityispalvelu Oy:n lisäksi. *Nero 2000* oli toki yrityksen tunnetuin tuote, mutta lisäksi myynnissä oli opetusohjelmia, kuten Commodore 64:lle *Bio-Syntax Method englanninkielen opiskelupaketti 1–6* ja yrityskäyttöön tarkoitettuja kieliohjelmistoja. Taisto tiimeineen ymmärsi pelillisyyden merkityksen opetuksessa jo varhaisessa vaiheessa, osaltaan tasoittaen tietä nykyisille opetuspeleille.

Pienessä yrityksessä melkein kaikki oli tehtävä itse, joten myyntiä varten levykkeet monistettiin suoraan tavallisella levyasemalla. Kasetit pystyttiin kuitenkin monistamaan asiaan paremmin soveltuvalla laitteella kloonaamalla master-nauhaa. Painotuotteet ja kotelo ostettiin muualta. Ohjelmat saatiin kauppaketjujen myyntiin, muun muassa Infoon. Myös *MikroBitin* artikkeli oli omiaan auttamaan yrityksen nimen tunnettavuudessa.

Alatyylisten ilmausten käyttö saa pöllönpoikasen pinnan kiristymään.

Nero 2000, sarjanumero #01, kaikkien levy-Nerojen isukki.



PC-versiossa pystyi hämään pomoa muka-taulukkolaskennalla.

Commodore 64 -version aloitusruutu ja herra nro 0647.

Neron kuriositeetteja

Moni on huomannut, ettei *Nero 2000* pidä alatyylisistä ilmauksista vastauksina. Pelin tekijöiltä löytyikin huumoria vastauksena pelaajan kiroilulle. Käyttäjän kirjoittaessa sopivan kirosanan peli viittaa humoristisesti ärtyneen pöllön kera toiseen yrityksen tuotteeseen, eli Bio-Slang-sanastoon (”*Englannin kieliohjelma NO:6, Bio-Slang*”).

Vastaus tosin hämmensi useimpia pelaajia, koska suoraan sen nimistä tuotetta ei ollut markkinoilla. Monet pitivät tätä ainoastaan vitsinä, mutta totuus on kuitenkin toinen. Tämä sanasto näet löytyy Bio-Syntax Methodin englannin opiskelupaketista 1–6, joka julkaistiin ainakin Commodore 64:lle, mahdollisesti myös IBM PC:lle ja MSX:lle. Tuotteesta löytyy kuusi erilaista sanastoa levykkeellä sekä pieneen kirjaseen painettuna. Yksi sanastoista on vanhahtavaan brittienglantiin pohjautuva Bio-Slang, mikä ei tosin käy yksiselitteisesti ilmi pakkauksen päältä.

Pakkauksessa todetaankin ”BIO-SYNTAX KIELENOPETUKSEN UUDET ULOTTUVUUDET, TÄYDENNÄ ENGLANNINKIELEN TAITOASI!”, mutta päälle liimatussa tarrassa ei ole ruksia Bio-Slangin kohdalla. Ei siis ihme, ettei sitä mystistä Bio-Slangia suorilta löytynyt, vaikka kyllä se siellä levyllä on.

Lohdutuspalkintona väärin vastaavia saattaa kiinnostaa pelistä löytyvä huijauskoodi, jolla vastauksen saa aina oikein. Mikäli haluat huijata Commodore 64 -versiossa, syötä $%&’x ja vastaus on aina oikein. *Neron* PC-versioon on taasen toteutettu F1-näppäimeen ”Pomo tulee” -toiminto, eli peli avaa DOS:n taulukkolaskentaa muistuttavan näkymän. Easter egg -tyyppisiä ominaisuuksia on siis ollut myös suomalaisissa peleissä jo kauan.

Tuntemattomat tekijät

Nero 2000:n tekijätiimi pysyi useimmille tuntemattomana pitkään. Tekijöiden henkilöllisyys pysyi pienen piirin tietona vuoteen 2016 saakka, vaikka moni on yrityksestä ja sen tuotteista jälkikäteen kysellytkin. Itselleni tiimin kokoonpano selvisi vasta käytyäni Timo Kokkosen luona noutamassa materiaalia Suomen pelimuseoon sekä arkistointia varten. Olin esimerkiksi ihmetellyt pitkään, miksi pelin alussa tekijän nimi oli vain numerosarja 0647. Selvisi, että Timo oli tehnyt nimimerkin syntymäajastaan laskimen heksamuunnoksen avulla. Hänen mukaansa moni muukin käytti vastaavan tyyppisiä nimimerkkejä aikanaan, sillä silloin ei osattu ollenkaan arvata, kuinka suuri juttu pelistä tulisikaan, eikä nimelle nähty arvoa.

IBM PC -version Timo toteutti 1987 joululomalla. PC ei tuohon aikaan ollut yksinkertaisin mahdollinen alusta graafisesti hienojen pelien toteuttamiseen, ja versioitakin olisi pitänyt tehdä useita. Niinpä näyttävyyteen ja musiikkiin ei PC-versiossa sen suuremmin panostettu, ja liiketoiminnallisista syistä peli toteutettiin merkkigrafiikoilla. Varhaiseen PC-versioon jäi myös bugi, joka sotkee tietyntyyppiset kysymykset mahdottomiksi vastata. Ei tosin ole selvää, tehtiinkö myyntiin useampia kuin yksi erä, koska Timon arkistoista löytyi useita eri päiväyksellä varustettuja versioita.

Itse peli herätti huomiota ulkomaita myöten: jopa Britannian yleisradioyhtiö BBC kutsui tiimin Lontooseen. Alkujaan matka oli palkintomatka johonkin kilpailuun, mutta lopulta siitä muodostui kirjakauppojen myyninedistämismatka, ja reissua varten saatiin British Airwaysin lentoliput. Mukana matkalla olivat Taisto Orre, David Cumberworth, Timo Kokkonen, Seija Hirvinen, Kaj Lassander ja eri kirjakauppojen edustajia. Sokerina pohjalla matkalla oli mukana myös Printti-lehden päätoimittaja **Reijo Telaranta**. Ulkomailla kilpailu oli kuitenkin todella kovaa, ja markkinoilla oli lukuisia kilpailevia tuotteita, joten Ison-Britannian valloitus olisi vaatinut vielä paljon lisää panostusta.

Myös Amstradia ja muita alustoja arvioitiin peliä varten, mutta niiden markkinat todettiin liian pieniksi. Ainoastaan Amiga-version toteuttaminen aloitettiin, ja peliin ehdittiin jo työstää parempia grafiikoita. Amiga-käännöksestä kuitenkin luovuttiin, ja yrityksessä päätettiin keskittyä pelkästään IBM PC:n kieliohjelmiin.

Vaiheet Neron jälkeen

Taiston mukaan tiimin kehitystaidot olivat hyvät, mutta liiketoimintaa ei valitettavasti hallittu riittävästi. Esimerkiksi markkinointiin panostettiin suuren yrityksen elkein, mutta tulovirta oli siihen nähden liian pieni. Vaikka *Nero 2000* myi hyvin, jakeluketjut ottivat suuren osan myynneistä. Jäljelle jääneestä osuudesta täytyi vielä maksaa palkkaa ja jakaa tuotot yrityksen osakkaiden kesken, joten lopulta viivan alle ei jäänyt paljoakaan. Neron menestyksestä huolimatta ammattikäyttöön tarkoitetut kieliohjelmat olivat Bio-Syntax Methodin parasta pääomaa, ja moni taho olikin niistä kiinnostunut.

> *”Kyllä se banaani tosiaan on kasvitieteessä marja, ei sen takia kannata koneelle manailla ja näppäimistöä murjoa.”*

Lontoon markkinointimatkan aikoihin Kaj Lassander halusi ostaa Bio-Syntax Methodin liiketoiminnan ja oli vakuuttunut tuotteiden menestymisestä markkinoilla. Kaj oli perustamassa osakeyhtiötä hankintaa varten, mutta valitettavasti hän menehtyi melko pian matkan jälkeen. Näin ollen suunnitteilla olleet yrityskaupat kariutuivat ja samaan aikaan taloudelliset paineet kasvoivat.

Promentor Solutions osti osan liiketoiminnasta, ja osa kieliohjelmista myytiin Superbrainille, eli Kasanen koulutusyhtiöille. *Nero 2000*:n oikeudet kuitenkin päätettiin pitää itsellä. Liiketoiminta jouduttiin kaikesta huolimatta ajamaan alas taloudellisten vaikeuksien tähden. Tämän jälkeen Taisto jatkoi Integro Finland Oy:n kouluttajana ja muut työntekijät siirtyivät eteenpäin. Velkajärjestelyjen kautta velat saatiin maksettua noin viidessä vuodessa. Kieliohjelmia ostaneen Superbrainin toiminta päättyi myös myöhemmin, joskin liiketoiminta jatkui toisen yrityksen piirissä. Toiminimi jäi kuitenkin elämään, mutta lopullisesti Bio-Syntax Methodin liiketoiminta päättyi 24.8.2017. Promentor Solutions toimii kuitenkin edelleen, ja Timo kehitti siellä muun muassa multimediaan perustuneen opetusohjelman, joka palkittiin 2000-luvun vaihteen seudulla.

Missä he vaikuttavat nykyisin?

Nykyisin 76-vuotias Taisto Orre on ehtinyt moneen ja teki muun muassa värikkään poliittisen uran. Hän toimii paluumuuttajana aktiivisesti SDP:n riveissä ja osallistuu edelleen mahdollisuuksien mukaan poliittiseen vaikuttamiseen Hyvinkäällä ja Facebookissa. Hän on opiskellut myös monia kieliä ja aikoo opiskella edelleen. Vierivä kivi ei sammaloidu, kuten sanotaan. Nykyisin noususs olevan lisätyn todellisuuden maailma kiinnostaa Taistoa erityisesti, ja hänellä onkin mielessään paljon uusia sen avulla toteutettavia ideoita. Pokemon Go- tai Walking Dead -tyyppiset pelit eivät tosin saa häneltä kiitosta. Taiston mukaan aika näyttää – jos hyvä tiimi vain löytyy, lisätyn todellisuuden ympärille ajateltua projektia voisi edistää.

Timo Kokkonen siirtyi Bio-Syntax Methodin toimintojen yrityskaupassa Promentor Solutionsille ja työskenteli

Taisto Orre ja Timo Kokkonen Suomen pelimuseon avajaisissa tammikuussa 2017.

siellä Teknillisen korkeakoulun opintojen ohessa. Nykyisin hän asuu vaimon ja kahden pojan kanssa Espoossa ja vastaa Nokialla ohjelmistojen tuotekehitysympäristöistä. Suomen pelimuseossa olevat *Nero 2000* -reliikit ovat Timon henkilökohtaisista kokoelmista. Hänellä on yhä tallessa laitteisto, jolla *Nero* alkujaan tehtiin, ja kehityksessä käytetyt levykkeet.

David muutti melko pian liiketoiminnan loppumisen jälkeen takaisin Britanniaan ja on tiettävästi nykyisin liikemies. Häneen ei saatu artikkelin osalta yhteyttä yrityksistä huolimatta.

Nero 2000:n paluu?

Hyvinkään *Aamupostin* tammikuun 2017 artikkelissa mainitaan, että pelin uusi versio tulisi ulos Suomen pelimuseon julkaisemana. Artikkelissa on kuitenkin virhe, sillä uutta julkaisua suunniteltiin digiarkeologiaan erikoistuneelta Kasettilamerit-ryhmältä. Idea pelimuseon versiosta jäi kuitenkin kolkuttelemaan mielen sopukoihin melkein vuodeksi, ennen kuin asia vieri eteenpäin.

Kaiken kaikkiaan *Neron* uudistetusta versiosta on keskusteltu demoskeneen liittyvissä porukoissa jo vuosien ajan. Tämä uusiointikeskustelu kuitenkin käynnistyi **Ville Joupin** ja **Aki Sivulan** ajatuksenvaihdosta, kun Nero 2000:een liittyvä salapoliisityö starttasi kesällä 2016. Taustalla olivat aiemmat keskustelut Nero 2000:n demoskeneversiosta, jota voisi vaikkapa käyttää ohjelmanumerona demopartyillä.

Asia otettiin uudelleen esiin noin vuosi alkuperäisen yhteydenoton jälkeen. Ehdotin asiaa jälleen Kasettilamereille ja keskustelin sekä Taiston että Timon kanssa. Keskustelimme asiasta myös Suomen pelimuseon tutkijan **Niklas Nylundin** kanssa, ja ehdotus sai kaikilta hyvän vastaanoton.

Yksi haasteista on uusien kysymysten luonti: kuka ne tekisi? Yksi hyvä vaihtoehto olisi kysymysten joukkoistaminen Suomen Pelimuseon kautta. Kysymysten neljä eri kategoriaa määrittelisi kuitenkin museo. Toistaiseksi projektin toteuttamisen laajuus jää vielä nähtäväksi, mutta luvat uuteen versioon kysyttiin ja saatiin sekä Taistolta että Timolta loka-marraskuussa 2017. Myös Pelimuseossa myytävä kaupallinen versio on sen myötä mahdollinen. Ylipäätään Kasettilamereilla on nyt myös lupa tehdä pienempiä customversioita erilaisiin tarkoituksiin, toki ilmaiseksi.

Alkuperäiset lähdekoodit ovat yhä tallessa Timo Kokkosella, joten uusien versioiden tuottaminen onnistuu varmasti. Periaatteessa Basic-laajennukseen tehtyjen koodien virittelykin onnistuu, mutta katsotaan onko tarvetta. Selvää on, että varsinaisesta koodien viilaamisesta ja kysymysten paketoinnista tulee vastaamaan Kasettilamerien **Jorma Oksanen**. Peliin on tarkoitus tehdä vähintäänkin sellainen korjaus, ettei samaa kysymystä kysytä uudelleen saman pelin aikana. Ehkä jotain muutakin pientä viilausta tehdään. Stay tuned…

Nero 2000

Tekijä: Bio-Syntax Method
Julkaisuvuosi: 1987
Alustat: Commodore 64, MSX, PC MS-DOS

LÖYDÄ OIKEA KUMPPANI TIETOSUOJA-ASIANTUNTIJAKSI

TOUKOKUU 2018

TIETOSUOJAMALLI

PARAS SÄHKÖINEN TYÖKALU TIETOSUOJA-ASIOIDEN HOITAMISEEN

Tarvitsetko apua tietosuoja-asioihin? Etkö ehdi hoitaa kaikkea itse 25.5.2018 mennessä? Capnovan tarjoama Tietosuojamalli on tehokas tapa ymmärtää mikä uudessa tietosuoja-asetuksessa on oleellista ja mitä toimenpiteitä se organisaatioltanne vaatii.

- Valmis etenemismalli kysymyksineen
- Helppo dokumentaation ylläpito
- Päivitys lakimuutosten mukaisesti
- Kattava raportointi

Anna meidän auttaa yritystäsi suojaamaan henkilödataa tietosuoja-asetuksen edellyttämällä tavalla!

Ota yhteyttä: myynti@capnova.fi, 020 759 7541
www.capnova.fi

Capnova

JOKSTORIAA

osa IV: kun kotimikro tuli taloon

Teksti: Jukka O. Kauppinen
Kuvat: Susanna Rantanen, Niila T. Rautanen,
Bitmap Books, FunStockRetro.co.uk

JOKstoriassa on junnattu jo ihan tarpeeksi mikrokivikauden tapahtumista. Nyt on aika tehdä aikahyppy taas eli kertoa kuinka Jukka viimein astui tietokoneaikaan.

En liene aivan ainoa junnu, joka on istunut lukematttomia hetkiä kavereiden luona pelaamassa ja näpräämässä heidän tekniikkavempeleitään ja pelikoneitaan. Sillä tavoinhan voidaan viettää mukavasti aikaa kavereiden kanssa, ja siinä sivussa säästytään monelta vaaralliselta asialta, kuten urheilulta ja ulkoilmalta. Ja mikäs sen kivempi tapa päästä kiinni näpertämiseen ja jännittäviin peleihin, jos omaa tietokonetta tai konsolia ei vielä ole.

Kaikilla ei ollut – oikeastaan aika harvalla oli noina jännittävinä 1980-luvun puolivälin aikoina. Mutta kyllähän VIC-20:t ja muut varhaiset mikrot yleistyivät hurjaa vauhtia. Vaikka kilpailu oli kovaa, niin Commodoressa oli jo jotain maagista vetovoimaa.

Kenties tähän väliin täytyy viskata yksi mihinkään liittymätön muisto. Luokkakaverini **Olli** oli reilu kaveri, jolla oli Atari 2600 -konsoli ja monta peliä. Oma konsoli oli yhä harvinaista herkkua, ja vaikka kone olikin jo iäkäs, eivätkä pelit olleet yhtä hyviä kuin esimerkiksi vuokraamassamme ColecoVisionissa, niin omalla konsolilla sai ainakin pelata koska vain halusi. Niistä ajoista minullekin jäi syvä rakkaus *Combat*-peliä kohtaan. Siinä tavoitettiin niin oivallisesti parin hassun kilotavun voimin kaksinpelaamisen riemut. Se oli niin kivaa, että pelikylään kannatti polkea kerta toisensa jälkeen 15 kilometriä. Suuntaansa.

Samoihin aikoihin toisella luokkakaverillani oli ZX Spectrum. 1982 julkaistu brittimikro oli suosittu myös Suomessa, vaikka moni onkin jälkikäteen dissannut Spectrumia ja sen pelejä. Tavallaan syystä, sillä olihan se aika kehno kone. Mutta edullinen, monipuolinen ja kyllä sille pelejä tehtiin!

Jälkikäteen katsottuna yllättävän moni arvostettu peliklassikko syntyi nimenomaan Spectrumilla, ja vain osa niistä käännettiin myös C64:lle. Olinkin oikeastaan aina onnekas, kun Jussi asui aivan meidän ala-asteemme vieressä. Ei muuten ollut ihan harva kerta, kun kiipesin koulun aidan yli ja hilpaisin pelaamaan Spectrumilla väli- tai ruokatunnin ajaksi.

Jussilla olikin monta hyvää peliä, joiden parissa oma pelimaailmani muovautui hyvällä tapaa. Ultimate the Gamen klassikot ja maailman ensimmäisiin lukeutuva 3D-toimintaseikkailu *Ant Attack*. Jokin jaetun ruudun lentosimulaattori. Siistiä oli sekin, että Spectrumille oli muutamia todella hyviä strategiapelejä, jotka ohjasivat minut hyvälle sotapelilinjalle.

Kopiokaverit

Ja eräänä päivänä minäkin sain lopulta oman koneen. Luultavasti sen hankinnasta oli keskusteltu jo pitkään, jo VIC-20-ajoista saakka. Vaikka koneiden spekseilläkin oli väliä, niin suurin merkitys oli kaveripiirillä. Kuusnelosia

löytyi useampia jo lähipiiristä, joten pelien saanti oli varmaa.

Eihän siinä ajassa ja iässä tekijänoikeuksista tiedetty ja vielä vähemmän välitetty, vaan pelien kopiointi laskettiin samaan kategoriaan kuin musiikin tallentaminen radiosta kasetille. Kaikkihan sitä teki.

Kuusnelosen kräkätyt pelit olivat tietysti tärkeässä asemassa koneen suosion suhteen. Vaikka myös alkuperäisiä kasettipelejä pystyi kopioimaan tupladekkimankalla tai -stereoilla, niin näppärät warettajat olivat kehittäneet aivan omia systeemejään. Siinä missä kuusnelosen alkuperäiset kasettipelit saattoivat latautua vartista tuntiin, niin kräkätyt pelit latautuivat minuuteissa. Niitä oli myös erittäin helppo kopioida. Peli vain ladattiin Turbo-ohjelmalla koneen muistiin, vaihdettiin kasettia ja tallennettiin omalle kasetille. Helppoa ja nopeaa!

Huhut kertoivat jopa, että meillä oli omalla pikkupaikkakunnallamme joku taitava tyyppi, **Kasper**, joka osasi kräkätä pelejä ihan itse. Hän oli jopa tehnyt aivan oman turbotape-systeemin. Mikä sankari!

Ja niin pelien saatavuus ja sosiaalinen houkutus tekivät meikäläisestäkin Commodore 64 -käyttäjän. Kyllä sitä hetkeä olikin odotettu. Muistan elävästi, miten luin kirjastossa tietokonelehtiä ja kaipasin, että kylläpä olisi uskomattoman hienoa pelata *Eliteä* ja muita uskomattomia eepoksia.

Miksei Elite toimi?

Tietokonekauppaahan käytiin noina 1980-luvun puolivälin vuosina aika lailla kaikkialla. Jopa kodinkone- ja musiikkiliikkeet pursusivat konepaketteja, eikä kaikilla ollut minkään valtakunnan osaamista tai ymmärrystä myymistään tuotteista. Joillakin ei ollut edes demokonetta, vain esitteitä ja kone näyteikkunassa.

Oma koneeni ostettiin jonkinlaisesta huolto- ja elektroniikkaliikkeen sekasikiöstä. Pikkuruinen puoti City-käytävän perällä oli näet siitä hyvä lafka, että siellä sentään tajuttiin koneiden päälle, ja kokeiltavana oli useita koneita ja uutuuspelejä. Junnulaumoja siis riitti. Hyvä palvelu johti myös kauppoihin, sillä saimme heiltä hyvän kone+kasettiasema-paketin lisäksi yhden vapaavalintaisen pelin sekä joystickin.

Muistan miten puntaroin pelivalintaani. Ensin valitsin *Summer Games II*:n, koska se oli aivan uusi, eikä sitä ollut vielä kavereillakaan. Sillähän saisi arvokkaita kamupisteitä, ja sitä kopioimalla voi vaihtaa itselleen lisää pelattavaa! Vaihtotalous oli näet voimissaan - jos sinulla ei ollut mitään annettavaa muille, niin olit kopiopiirin pahnanpohjimmainen. Sait uusia pelejä vain armopaloina ja silloinkin viimeisenä.

Vaan eipä *Summer Gamesistä* tullut ekapeliäni, sillä viime metreillä muutin mieltäni. Haluan sittenkin pelin, joka on enemmän omaan makuuni, jota oikeasti himoitsen ja josta olen lukenut niin paljon. *Eliten*.

Ja se oli hyvä päätös se. Viihdyin *Eliten* parissa lukemattomia tunteja. Tosin oikein kunnolla vasta sen jälkeen, kun eräs ongelma oli ylitetty. Kaupastahan haettiin pelkkä kone, joten se piti lyödä kiinni perheemme olohuoneen televisioon. Tämä rajoitti pelitunteja, sillä noina kasarin ihmeellisinä vuosina televisio oli tärkeä osa arkipäivää. Kun päivällä oltiin koulussa ja illalla katsottiin uutiset, päivän leffa ja *Dallasia*, *Taisteluplaneetta Galacticaa*, *Ihmemiestä*, *Ritari Ässää* tai muita välttämättömyyksiä, niin ei siinä ylettömästi jäänyt läksyjen ja muiden puuhastelujen jälkeen aikaa tietokonehommille. Joskus veto oli niin kovaa, että edes vieraat eivät saaneet haitata pelisessiota. Siinä riitti vanhemmalle väelle ihmeteltävää, kun tv:stä tuli muutakin kuin kolmen virallisen kanavan ohjelmaa. Jossain vaiheessa sentään sain oman matkatelkkarin ja olkkari vapautui tv-kulttuurille.

Loppukevennys: mitä tapahtui, kun löin oman koneeni ensimmäistä kertaa telkkariin, säädin kanavat kohdalleen ja latasin pelin? Ei oikein mitään. *Eliten* upea vektoriavaruusalus pyöri ruudulla, mutta peli ei alkanut, alus ei lentänyt. Onko tikku vai kone rikki? Lopulta soitin huolissani kaverilleni, joka osasi kanavoida tukihenkilöviisauttaan ja antoi viisaan ohjeen:

”Oletko kokeillut laittaa joystickiä toiseen tikkuporttiin?”

Sehän se. *Eliten* peluu saattoi alkaa. Näin amatööri minä olin, mutta en kauaa. Tästä tulikin hyvä elämän ohjenuora: debugatessanne ongelmia tarkistakaa aina ensin johdot ja kokeilkaa myös muita liittimiä.

JOKstorian seuraavassa osassa: QuickShot II ja vedonlyönti.

Jukka O. Kauppinen alias Grendel/Byterapers on pitkän linjan toimittaja, joka kirjoitti ensimmäisiä tekstejään mekaanisella kirjoituskoneella. JOKin ensimmäinen peliarvostelu julkaistiin 1986, mistä lähtien hän on ahkeroinut tauotta kirjoittaen juttuja muun muassa videopeleistä, tietotekniikasta, viihteestä, ilmailusta ja burleskista. JOKstoriaa-sarjassa muistellaan niin kotimikroilun kuin peli- ja digijournalismin maailmaa grennulasien läpi nähtynä. jukka@skrolli.fi

TEKNIIKAN AKATEEMISET

TEKIN JÄSENYYS KANNATTAA!

TEKin jäsenenä työelämän vakuutusturvasi on kunnossa. Saat paljon rahanarvoisia etuja ja palveluja. Asiantuntijamme tuntevat parhaiten tekniikan akateemisten työelämätilanteet ja palkat. He ovat käytössäsi ja neuvottelevat puolestasi, jatkuvasti.

www.tek.fi/jasenedut

PALKKANEUVONTA

Palkkanosturi
Voit tarkistaa, paljonko palkkaa kannattaa pyytää. Palkkanosturi piirtää palkkakäyrät muuttujavalintojesi mukaan. Pohjatietona on vuosittain jäsenten keskuudessa tehtävä työmarkkinatutkimus.

Palkkasuositukset
jotta harjoittelu-, diplomityö- ja alkupalkat olisivat oikealla tasolla!

Henkilökohtainen palkkaneuvonta
sähköpostitse tai puhelimitse.

Ulkomaan palkat
tutkitaan jäsenten keskuudessa joka kolmas vuosi. Palkkaneuvojillamme on käytössään lisäksi kansainvälisiä palkkatilastoja.

VAKUUTUKSET

Työttömyysturva
IAET-kassan jäsenyys sisältyy jäsenmaksuun. (Palkansaaja-asemassa olevat täysjäsenet ja kassaan liittyneet, työssä käyvät opiskelijat)

Vastuu- ja oikeusturvavakuutus
sisältyy jäsenmaksuun ja kattaa oikeudenkäyntikuluja työsuhderiidoissa, työssä sattuneita henkilö- ja esine- vahinkoja sekä varallisuusvahinkoja. (Palkansaajat, yksinyrittäjät ja opiskelijat)

Alennuksia vakuutuksista
OP Vakuutukselta mm. jatkuva 20 % alennus matkustajava- kuutuksesta sekä 5 % alennus matkatavaravakuutuksesta. Ifin henkivakuutus TEKin jäsenille on yksi Suomen edullisemmista.

5 %

ALENNUSTA 20 %

URAPALVELUT

CV-klinikka
Räätälöityjä CV-ohjeita ja henkilökohtaista* CV-sparrausta uravalmentajiltamme.

Työnhaku
Henkilökohtaista* valmennusta kun olet aktiivisessa työnhaussa. Kattavasti tietoa ja työnhaun välineitä mm Haastis-sovellus, kun haluat harjoitella työhaastatteluvastauksia ja GoinGlobal-palvelu ulkomaille lähtemisen tueksi.

Uravalmennus
Henkilökohtaista* valmennusta, kun haluat pohtia omaa uraasi ja sen suuntaa.

TEKrekry LinkedIn-ryhmä
Avoimia työpaikkoja, verkottumista ja urakeskustelua.

* Henkilökohtaista valmennusta tarjoamme täysjäsenille sekä valmistumisvaiheessa oleville. Uravalmentajamme auttavat kampuksilla opiskelijoita aktiivisimpaan kesätyöhakuaikaan erilaisissa tapahtumissa, luennoilla ja työpajoissa.

JULKAISUT JA LEHDET

Sähköinen jäsenkirje
ajankohtaista hyötytietoa sähköpostitse kerran kuussa.

Alma Talentin ammattilehdet
Täysjäsenenä saat kaksi valitsemaasi Alma Talentin ammattilehteä joko diginä Summa-palvelussa tai perinteisesti kotiin kannettuina. Opiskelijajäsen saa Tekniikka&Talouden ja Talouselämän diginä.

TEK-lehti
5 numeroa/vuosi, sisältönä tekniikka, työ ja teekkarihenki.

TEK-verkkolehti
uutisia ja taustoja lähes päivittäin printtilehtien välillä.

Teekkarin työkirja
Opiskelijan paras työnhakuopas.

TEKin taskukalenteri tai seinäkalenteri
Maksuton, mutta tilattava jäsenetu.

Alma Talentin kirjoista
30 %:n alennus.

ALENNUSTA 30 %

LAKIPALVELUT

Työsuhdeneuvonta
Juristimme neuvovat sinua henkilökohtaisesti työelämän lakiasioissa. Saat mm. apua työsuhderiitoihin ja työsopimusten laadintaan.

Lakitietopankki
Paljon työelämän lakitietoa verkossa.

eLakimies
Vuorovaikutteisessa verkkopalvelussa saat vastauksia tavallisimpiin työsuhdekysymyksiin.

Perhe- ja perintöoikeudellista neuvontaa
lakimiehiltämme.

SINULLE, YRITTÄJÄ!

TEKin Yrittäjäklubi
toimii yrittäjien ja yrittäjähenkisten jäsenten yhdyssiteenä.

Henkilökohtaista oikeudellista neuvontaa
yrittäjyyteen liittyvissä asioissa.

Yrittäjän palvelutarjotin verkossa
Keskeistä yrittäjyyteen liittyvää tietoa aihealueittain, myös ulkoisilta yhteistyökumppaneilta.

Webinaareja yrittäjille
TEKin, Suomen Ekonomien ja Fondian yhteistyönä.

Virtuaalilakimies
Fondian verkkopalvelu tarjoaa yrittäjille lakitietoa liiketoiminnan tueksi.

Suomen Yrittäjien laaja palvelupaketti
TEKin yrittäjäjäsen voi hankkia Suomen Yrittäjien laajan palvelupaketin hintaan 90 euroa vuodessa.

UKKO.fi
Laskutuspalvelu TEKin jäsenille 20% alennuksella.

ALENNUSTA 20 %

TAPAHTUMAT JA VERKOSTOITUMINEN

- **Uraillat**
  TEKin täysjäsenille tarkoitettuja tilaisuuksia, joissa käsitellään työuraan, työnhakuun, työhyvinvointiin ja työelämässä jaksamiseen liittyviä teemoja. Osa urailloista striimataan ja tallennetaan, joten niitä voi seurata verkossa.
- **Nuorten osaajien afterit**
  Verkostoitumista ja uratarinoita työelämässä aloitteleville ja opintojaan viimeisteleville tekkiläisille.
- **Popup-tapahtumat kampuksilla**
  tuovat TEKin asiantuntijat lähelle opiskelijoita.
- **Koko perheen tiedepäivät**
  Innostus tekniikkaan syttyy itse tekemällä. Tiederasteilla lapset pääsevät kokeilemaan tekniikan soveltamista.
- **Alueseurat ja kerhot**
  tarjoavat mahdollisuuden verkostoitua ja harrastaa hyvässä seurassa.
- **Satoja tapahtumia vuosittain ympäri Suomen**
  Urailtoja, jäseniltoja, opiskelijoiden tilaisuuksia, perheiden tiedepäiviä, koulutuksia, webinaareja, kerhojen ja alueseurojen tapahtumia. Tapahtumakalenteri www.tek.fi etusivulla.
- **TEK Lounge**
  Ensimmäinen TEKin jäsenten oma etätyö-, tapahtuma- ja verkostoitumistila on avattu Tampereen teknillisen yliopiston Tietotaloon.

PANKKI- JA SIJOITUSETUJA

Jäsenetuja asuntolainaan
Danske Bank tarjoaa TEKin jäsenille etuja asuntolainan marginaaliin, asuntolainan ja lainanlyhennysvapaiden toimitusmaksuun sekä päivittäiseen pankkiasiointiin. Kilpailuta asuntolainasi nyt!

Etulaina
Danske Bankin Etulainaa voivat hakea kaikkien pankkien asiakkaat helposti ja nopeasti verkossa ilman avausmaksua erittäin kilpailukykyisellä korolla.

Pankkipalvelut vastavalmistuneille
Danske Bank tarjoaa TEKin vastavalmistuneille jäsenille (valmistumisesta maksimissaan kaksi vuotta) etu- ja palvelukokonaisuuden, jonka arvo voi ylittää jopa 700 euroa vuodessa.

Rahastosijoituksia
ilman merkintä- ja lunastuspalkkiota Evli Pankissa.

Sijoittajan Tietopaketti
Osakesäästäjien Keskusliitto tarjoaa TEKin jäsenille maksuttoman Sijoittajan Tietopaketin vuodelle 2018.

EDUNVALVONTA

1. **Luottamusmiehet**
   Työpaikoilla henkilöstöryhmäsi valitsemat luottamusmiehet edustavat sinua ja kaikkia akavalaisia työsuhdeasioissa sekä auttavat työ- ja virkaehtosopimusten tulkinnassa. Työpaikoilla etujasi valvovat lisäksi myös yritysyhdistykset ja yhteyshenkilöt TEKin ja YTN:n asiamiesten tukemina.
2. **Omaluottamusmies.fi -verkkopalvelu**
   Työelämän tietopalvelu korkeasti koulutetuille. Tarjolla tietoa esimerkiksi työsopimuksesta ja palkasta.
3. **Työsuhteen ehdot**
   Henkilökohtaisen työsopimuksen lisäksi palkansaajien yleisistä työsuhteen ehdoista ja palkankorotuksista neuvotellaan työntekijä- ja työnantajaliittojen kesken. TEK neuvottelee työ- ja virkaehtosopimuksista osana akavalaisia neuvotteluorganisaatioita YTN:ää ja JUKOa. Neuvottelujärjestöt tukevat myös paikallisessa sopimisessa.
4. **Koulutus ja työvoimapolitiikka**
   TEK vaikuttaa koulutus- ja työvoimapolitiikkaan jotta sekä jäsenelle että koko Suomelle syntyisi kilpailukykyä korkeasta osaamisesta.
5. **Teknologiapolitiikka**
   TEKin tavoitteena on tekniikan tekijöille ja teknologiayrityksille suotuisa elinkeinopolitiikka.

EIKÄ SIINÄ VIELÄ KAIKKI!

Member+ -jäsenetuportaali
Akavalaisten liittojen yhteisessä portaalissa alennuksia muun muassa matkoista, harrastuksista, autoilusta ja hyvinvoinnista.

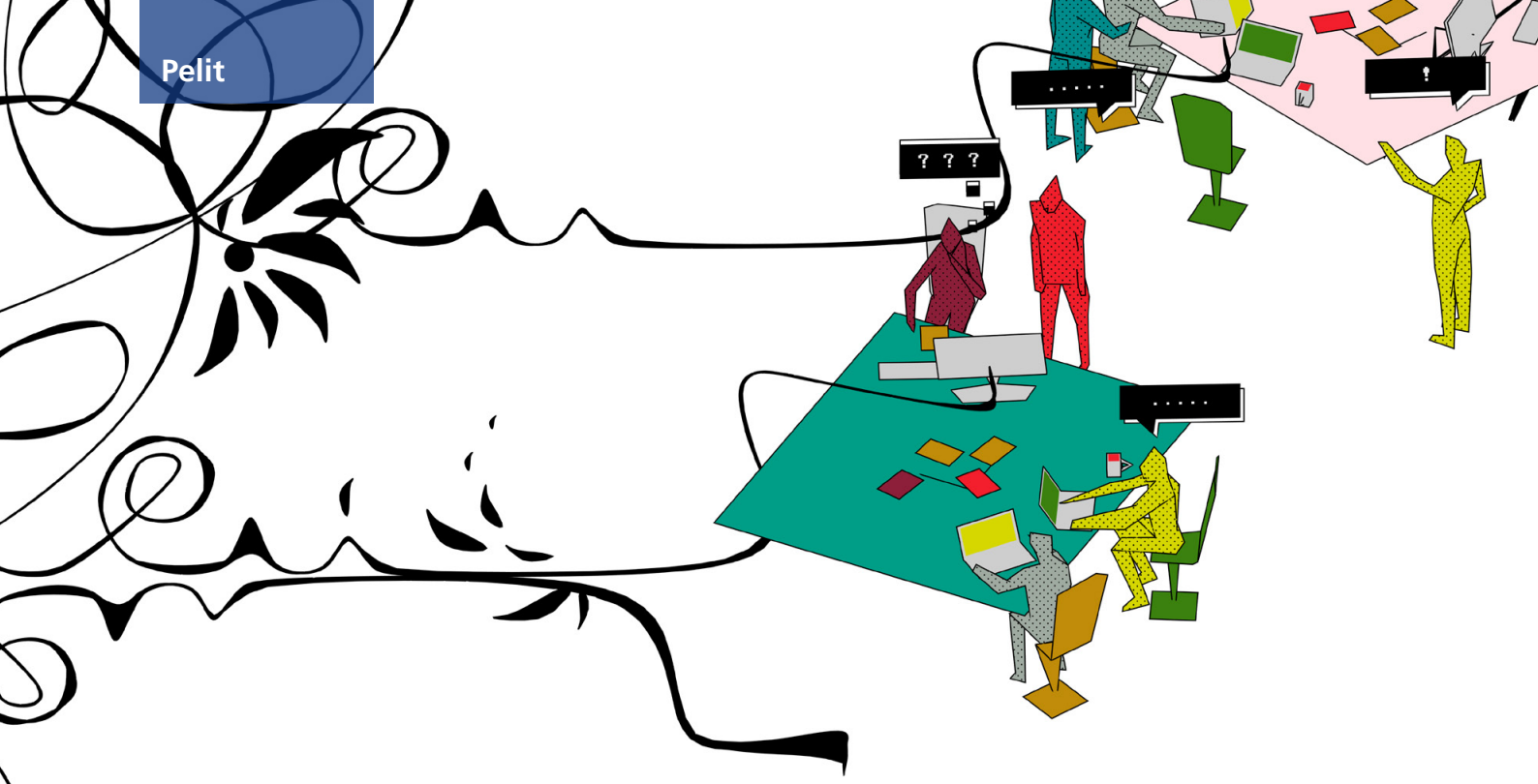
A-lomat
Tuettuja perhe- ja terveyslomia, hostellietuja ja edullista lomamökkien vuokrausta eri puolilta Suomea A-lomien kautta.

DI-sormukset
DI-sormuksen voi hankkia ainoastaan TEKin sormustodistuksella; jäsenille 50 euron alennus.

Ohjelmoinnin verkko-opiskelupaketit
- 20% Viope Solutions Oy:ltä.

Psykologin palvelut netissä
-10 % Therapionilta.

50 € ALENNUSTA 20 % 10 %



Mielikuvitus kuplii Game Jameissa
Ideatehdas sykkii pelin 48 tunnissa

Yksi viikonloppu, jonka aikana ideoidaan ja rakennetaan peli tyhjästä. Mikä saa ihmiset puskemaan itsensä äärirajoille pelintekemisen parissa, ja mitä jamikulttuuri tarjoaa pelialalle?

Teksti: Heidi Akselin ja Tiina Malin
Kuvat: Heidi Akselin, Toni Kortelahti, Jukka O. Kauppinen, Mikko Karsisto

Global Game Jam -tapahtumia on järjestetty vuodesta 2009 ympäri maailmaa. Niihin on osallistunut satoja tuhansia ihmisiä ja niissä on tehty tuhansia pelejä. Suomi liittyi mukaan jammailumaihin vuonna 2010, kun ensimmäinen Finnish Game Jam näki päivänvalon vain vuosi ensimmäisen Global Game Jamin jälkeen.

Tietokonepelifestivaali Assemblyn ohjelmistoon on vuodesta 2013 kuulunut Assembly Game Jam, kaikille peliharrastajille avoin pelien tekemiseen suunnattu tapahtuma. Jamit tulivat mukaan Assemblyihin Finnish Game Jam -yhdistyksen varapuheenjohtaja **Vesa Raudasojan** ideasta.

Raudasoja muistuttaa, että aiemmin Assyille tultiin demojen takia, mutta sittemmin pelaaminen nousi tapahtuman suosituimmaksi viihdykkeeksi.

”Pelien tekeminen on haluttu tuoda festivaaleille takaisin ”back to the roots”-meiningillä”, hän sanoi. ”Jameissa tuodaan yhteen niitä Assembly-kävijöitä, jotka eivät ole ennen kokeilleet pelintekemistä ja vastavuoroisesti pelintekijöitä, jotka eivät ole aikaisemmin tutustuneet tapahtumaan – kaikki voittavat.”

”Pelijamien perusidea on, että on tietyn ajan sisällä, Assemblyjen tapauksessa 62 tunnissa torstai-illan ja sunnuntaiaamun välillä, tehdään peli alusta loppuun”, jatkaa yhdistyksen puheenjohtaja **Annakaisa Kultima.** ”Messukeskuksessa voi myös yöpyä mikäli niin toivoo. Uni tulee usein tarpeeseen, sillä pelintekeminen on kognitiivisesti haastavaa. Monesti kokeneet jammailijat tuntevat rajansa ja osaavat jättää ’all-nighterit’ väliin, koska energiaa tarvitaan myös niille viimeisille tunneille.”

Jameissa syntyi tällä kertaa kymmenkunta peliä noin 25 osallistujan toimesta. Kultima itse jammaili kahdessa eri tiimissä. Pelinikkaroinnille oli varattu oma itsenäinen alue pelihallin keskeltä, jonne jammailijat saattoivat mennä ideoimaan yhdessä.

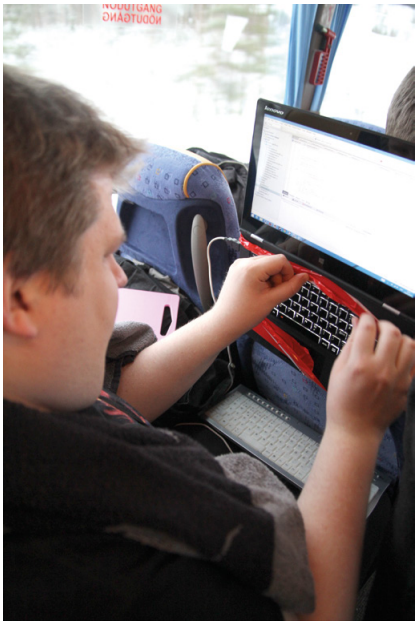
”Osa jammaa pöydässä, osa omilla paikoillaan”, Kultima sanoo.

Kultima ja Raudasoja kertoivat Suomen olevan ”kova jammailumaa”. Useat eri tahot järjestävät jameja, ja absoluuttisten numeroiden mukaan Suomi on kuudenneksi tuotteliain maa melkein sadasta. Väkilukuun suhteutettuna ainoastaan saarivaltio Guernsey vie voiton Suomesta. Kultima kertoo Suomessa olevan ”superjammereiksi” kutsuttuja henkilöitä, jotka ovat jammanneet yli 25 tapahtumassa, parhaimmillaan matkustaen ympäri maailmaa kansainvälisissä tapahtumissa. Etenkin Global Game Jam on harrastajille tuttu tapahtuma.

”Tämän vuoden tammikuussa Global Game Jamiin otti osaa yli 36 000 osallistujaa 95 eri maasta. Samalla nimeämisperiaatteella perustettu Finnish Game Jam on sekin kerännyt suosiota. Vuonna 2017 Global Game Jamiin osallistui Suomesta 22 eri paikkakuntaa ja joissakin kaupungeissa, kuten Helsingissä, oli useita toimipisteitä. Jameissa syntyi kaikkiaan 222 peliä 888 osallistujan voimin.”

Rakkauden jamit mahtuvat taskuun

Jamikansa kokoontuu muulloinkin: esimerkiksi syksyllä PocketGamer-Connects-konferenssissa järjestettiin 30 tunnin jamirutistus. Pocket Jam toi oman värinsä mobiilipelikonferenssin ohjelmaan. Jamien teemana oli *More*



Bussijamit oli haastava ympäristö pelien kehittämiseen. Ergonomia oli hitusen haussa, mutta onneksi linja-auton ikkunat toimivat kätevinä valkotauluina muistiinpanoille.

love less hate, jonka innoittamana 40 jammaajaa synnytti 11 peliä, joista kolme parasta palkittiin matkalla Pocket-GamerConnects Lontooseen. Jamit voivat siten myös edesauttaa työllistymistä pelialle, mutta Pocket Jameilla muistettiin silti pitää myös hauskaa.

Jamikonkari Katri Laine työskenteli Pocket Jameilla ja muistelee tilaisuutta tyytyväisenä: ”Oli tosi siistii nähdä, miten monet lähtevät jameihin ennakkoluulottomasti. Soppa, joka jameissa keitetään, on täynnä tulevaisuuden staroja.”

Jamit herättivät kiinnostusta myös konferenssikävijöissä ”Oli kiva nähdä, miten kiinnostuneita ohikulkijat on, että hei mikä tämä tapahtuma on. Kiinnostusta löytyi.” Katri kertoo reaktioista.

PETO IRTI

Pelijamit ovat ensisijaisesti ideatehdas, joissa saa päästää luovuutensa valloilleen ilman sen kummempaa maalia. Tärkeintä on antaa pelintekijöille vapaus luoda ja kokeilla mitä tahansa, ja mahdollisuus kehittää omaa osaamistaan, myös oman mukavuusalueensa ulkopuolelta. Kultima korosti jamien tärkeyttä myös verkostoitumisen kannalta.

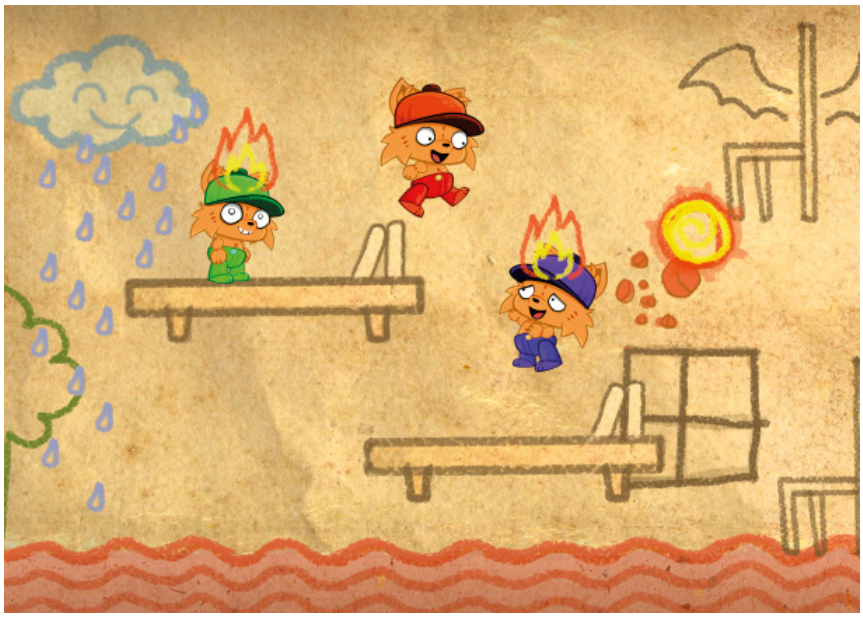
”Peliyrityksen syntyyn tarvitaan ihmisiä, joilla on toisiaan täydentäviä taitoja. Jos et pysty tekemään peliä yksin alusta loppuun, niin jamit ovat loistava mahdollisuus päästä tutustumaan uusiin ihmisiin ja löytää puuttuvien taitojen osaajia. Toki jameilla voi olla myös huonojakin kokemuksia ja siinä missä voit löytää potentiaalisia yhteistyökumppaneita, voit myös törmätä ihmisiin, joiden kanssa tiedät ettet halua ikinä työskennellä.”

”Jokaisesta jamista oppii aina jotain”, Raudasoja jatkoi. ”Oppii kommunikaatiotaitoja tiimissä, nopeampia ja tehokkaampia työskentelytapoja ja rytmittämään unentarvetta. Olit sitten ekaa kertaa liikenteessä tai sarjajammaaja, niin jokainen viikonloppu antaa jokaiselle jotain.”

”Monet tulevat jameille avoimin mielin, koska heillä on kokemusta miten eri jameilta on löytänyt erilaisia asioita. On löytänyt jonkun hyvän tyypin, oppinut uusia asioita, keksinyt kehityskelpoisia ideoita — on monia tasoja sille, mitä jameista voi saada. Osa jammailee mielellään tuttujen kanssa. Toisista tärkeintä se, ettei varsinaisesti tule kenenkään kanssa, vaan löytää uusia tyyppejä, jotka saavat omat aivot menemään ihan uuteen suuntaan”, sanoi Kultima.

Sekä Raudasoja että Kultima totesivat, että vaikka jamit toimivat parhaiten harrastuspohjalta, niihin ei kannata mennä opettelemaan pelintekoa nollasta lähtien.

Idearikkauden ja kekseliäisyyden ruokkimiseksi kaupallisuuden paineet sekä kilpailuasetelma on otettu pois. Tällä halutaan tukea sitä, että osallistujat suhtautuvat tapahtumaan henkilökohtaisena haasteena, eikä kukaan pelaa varman päälle yrittääkseen miellyttää esimerkiksi mahdollisia sponsoreita. Toki jameista voi saada myös potentiaalisia ideoita, ja hyvästä



Annakaisa Kultima ja Vesa Raudasoja luotsaavat Finnish Game Jamia.

raakileesta voi saada työstöllä seuraavan hittipelin. Myös pelialalla toisto ja yrittäminen ovat menestyksen rakennuspalikoita ja siihen tarvitaan sekä onnistumisia, että epäonnistumisia.

”Jokainen jameilla tehty peli myös julkaistaan tuloksesta riippumatta, mikä antaa pelintekijöille haastetta. Kaikenlaisen luovan työn tekijöille kehittyy itsekritiikkiä, mikä johtaa asioiden kesken jättämiseen ja arvostelua vaille jäämisen kautta kehityksen loppumiseen”, Kultima kertoi.

”Esimerkiksi Assembly-jameilla tehtaillut pelit päätyvät itch.io-nimiselle verkkosivustolle, mistä kuka tahansa voi ladata ne omaan käyttöönsä.”

Laine kannustaa jameista ja pelien tekemisestä kiinnostuneita: ”Rohkeasti mukaan! Yksin lähtiessä voi jännittää, mutta porukka on tosi vastaanottavaa. Rinta rottingille ja lähde!”

Mitä jammaillaan ja missä?

Ajan hengen mukaisesti jameilla tehdään eniten digipelejä, mutta toisinaan syntyy myös perinteisiä lauta-, roolitai korttipelejä – välillä hybridejäkin. Jotkut pelikehittäjät kokeilevat mielellään projekteja, joissa peliä saatetaan pelata ilman näyttöä auditiivisten ominaisuuksien varassa, tai joissa otetaan mukaan kineettisiä ominaisuuksia, kuten esimerkiksi liikkuminen ruudun läheisyydessä. Jamien järjestäjistä riippuen myös pelintekovälineet voivat olla kokeellisia, osallistujien kykyjen rajoissa.

”Game jamit ovat hyviä paikkoja kokeilla vaikka vaihtoehtoisia ohjaimia — ohjaimen voi rakentaa vaikka perunasta ja porkkanasta tai vesiämpäristä. Tänä päivänä on paljon harrastelijaelektroniikkaa, jota hyödyntämällä on helppoa lähteä kokeilemaan juttuja”, Kultima valotti.

”Joku vuosi takaperin Helsingissä järjestettiin vekotinjamit! Osallistujille annettiin kasa erilaisia leikkikaluja ja ohjeistettiin menemään, tekemään ja keksimään, miten ne toimii”, kertoi Raudasoja. ”Ensin hoksattiin palikoiden käyttötarkoitus ja sen jälkeen alettiin tekemään peliä.”

Entä millainen sitten on hyvä peli? Kultima ja Raudasoja ovat molemmat sitä mieltä, ettei siihen ole selkeää kaavaa. He korostavat, että jokaiselle pelille on oma markkinansa ja käyttötarkoituksensa.

”Peliä voi arvostella vain sen pohjalta, mitä on pyritty tekemään ja mitä kukakin arvostaa”, Kultima muistutti. ”Pelille tärkeää voi olla esimerkiksi helppo pelattavuus, ja jos siinä on paljon bugeja, se saattaa tienata pelille miinuspisteitä jonkun silmissä. Pelistä voidaan kuitenkin myös tarkoituksella tehdä mahdollisimman vaikeasti pelattava – aina on mahdollista rikkoa joku arvo, jota ihmiset laajasti tavoittelevat. Voisi ajatella, että jos pelin pelaaja tietää täsmälleen, mitä odottaa peliltä ja mitä hänen on siinä tehtävä, peli on hyvä. Kuitenkin myös esimerkiksi mahdolliset näkymättömät pikselit ja odottamattomat esteet voivat saattaa pelaajan miettimään, kuka pelin on tehnyt, ja miksi hän haluaa kiusata pelaajiaan. Näinkin syntyy suhde pelin pelaajan ja tekijän välille, mikä on arvokasta.”

Myöskään jammailun lokaatio ei ole kiveen hakattu. Haastetta tekemiseen voi hakea äärimmäisistä olosuhteista tai humoristisista häiriötekijöistä. Pelijameja on järjestetty muun muassa planetaarioissa, Suomea kiertävässä bussissa ja Lapin luonnossa ilman sähköä, juoksevaa vettä tai vakaata internetiä.

”Game jamit soveltuvat projekteihin, joiden järkevyydestä tai onnistuvuudesta ei aina olla ihan varmoja. Kaupallisuuden paineitahan ei ole lainkaan, eikä peliä tarvitse myydä”, sanoi Kultima.

”Ollaan haluttu rikkoa rajoja ja etsiä uusia: missä voi jammata, millä tavalla ja millä välineillä voi jammata,” Raudasoja jatkoi.

”Hurjimmassa jamissa oli kaksi osallistujaa. Samuli ja Jonne pyöräilivät tandempyörällä kuusi päivää Tukholmasta Malmöseen niin, että Samulilla oli ruutu ja näppis kiinnitettynä pyörän tankoon. He itse generoivat oman sähkönsä: Jonne ohjasi ja Samuli teki peliä”, Kultima kertoi nauraen.

Jamit ja maailmanrauha

Teknologian riemuvoiton myötä pelikulttuuri on jatkanut laajenemistaan. Nykyään pelaaminen on harrastus, joka kuuluu kaikille iästä, asuinpaikasta tai elämäntyylistä riippumatta. Kultima kehui pelijameja maailman parhaaksi harrastukseksi. Raudasoja komppasi ottamalla esiin arvokkaan näkökulman pelijamien merkityksellisyydestä ympäri maailman.

”Voivatko pelijamit pelastaa maailman? Maailmassa on paljon ihmisiä, joita politiikka ei kiinnosta, mutta he elävät esimerkiksi alueilla, joissa poliittinen ilmapiiri on painostava ja he joutuvat elämään sen kanssa. Tätä vastaan pelien tekeminen voi olla kovempi voima. Pelinkehittäjien löytäessä toisensa maailman tilanne unohtuu, kun keskitytään luomaan yhdessä jotain kivaa.”

”Pelin tekeminen on pienissä ryhmissä työskentelyä, ja siinä onkin sen voima. Useimmat pelintekijät eivät pysty tekemään pelejä yksin. Se pakottaa ihmiset yhteen, ja verkostoitumalla toisten paikallisten pelintekijöiden

Pelijameissa väki kokoontuu yhteen luomaan jotain uutta, ehkä erikoistakin. Päällisin puolin tapahtuma näyttää kuin miltä tahansa laneilta tai demopartyltä.

Survival Mode -jameissa lauma pelikehittäjiä vietiin keskellä ydintalvea Lappiin pikkuiselle erämökille. Lunta ja jäätä riitti loputtomiin, mutta säännöstelylistalla oli niin auringonvalo kuin sähkökin. Jopa energiankulutus täytyi optimoida.

kanssa voi löytää yhteisen sävelen ja sen, mitä haluaa tehdä. Jotkut pelikehittäjät kampanjoivat aiheesta sotien keskeltä ja jakavat omia tarinoitaan", Kultima jatkaa.

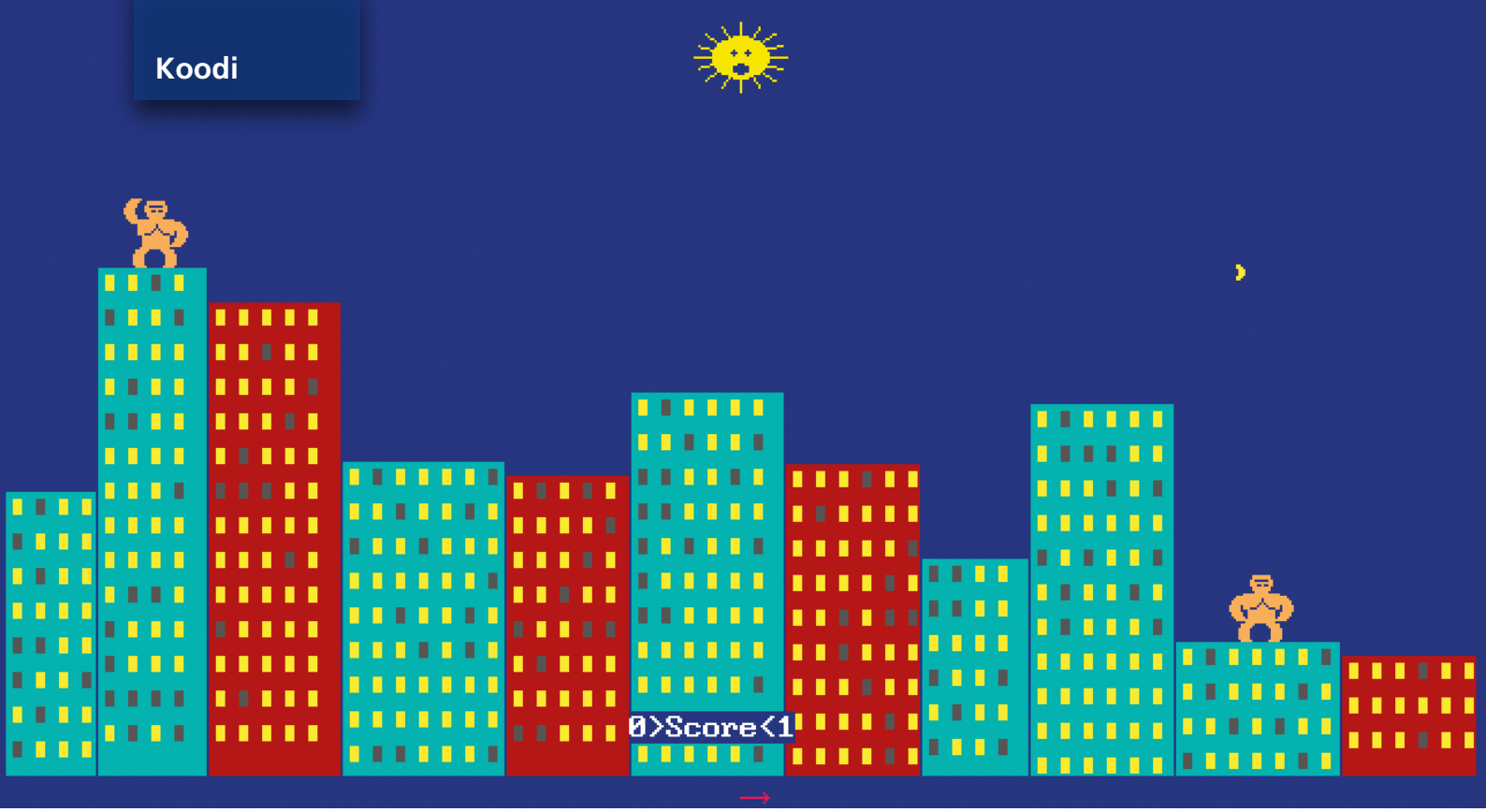
Jameja ja pelejä voi tehdä myös tieteellisten ongelmien ratkaisemiseksi tai johonkin asiaan liittyvän tietoisuuden levittämiseksi.

"Sellaisetkin pelijamit on pidetty kuin 'Boob Jam', joiden tarkoituksena oli rakentaa pelejä, jotka lisäsivät tietoutta rintasyövästä. 2018 keväällä järjestetään Saame Game Jam lisäämään tietoutta saamen kielen olemassaolosta ja saamelaisesta kulttuurista ja pyritään emansipoimaan paikallista nuorisoa tekemään pelejä omasta kulttuuristaan", Kultima maalaili.

Raudasoja painotti, että Finnish Game Jam haluaakin tuoda Assemblyille pelintekemisen iloa pelaamisen keskelle.

"Ei pelkkää Counter Strikea tai Quakea, vai mitä nykynuoriso nykyään pelaakaan!" hän nauroi.

"Toivottavasti he löytäisivät myös tietoa siitä, miten pelejä tehdään."



Olin teini-ikäinen QBasic-ohjelmoija

Lukuisten nykypäivän koodarien ensimmäinen kosketus ohjelmointiin tapahtui QBasic-nimisessä näppärässä pikku softassa, joka yhdisti helppokäyttöisen editorin ja ohjelmakoodia tulkkaavan ajoympäristön. Millainen ilmiö QBasic oli?

Teksti: Otso Kassinen Kuvat: Otso Kassinen, Janne Sirén

Tapahtui 1990-luvun puolivälin paikkeilla. Tarkkaa vuotta ei ole merkitty aikakirjoihin, joskin lehtiarkistoja penkomalla saattaisi syntyä jonkinlainen arvio päivämäärästä. Selasin nimittäin MikroPC:n numeroa, jonka kysymyspalstalla joku tiedusteli, miten PC:n piipperikaiuttimesta saisi ääntä ulos. Vastauksen mukaan helpoin keino olisi kirjoittaa QBasic-kielinen ohjelma, joka käyttäisi PLAY-lausetta.

Tätähän piti kokeilla heti, kun pääsin koneen äärelle. Siitä se vuosia kestänyt QBasic-ohjelmointiharrastus tai -elämäntapa sitten lähti. Muistan alkutaipaleella myös lukeneeni kirjastosta löytynyttä mikrotietokoneiden BASIC-kielten yleisopaskirjaa, joka oli painettu 1980-luvun puolella, mutta en ole varma, tapahtuiko tämä ennen MikroPC-valaistumista. Joka tapauksessa musisointimahdollisuus oli se asia, joka räjäytti pankin ja tuotti tunteen, että ohjelmointi antaa avaimet taianomaisen mukaansatempaavaan, käytännölliseen tekemiseen tietokoneella.

Kuten mikä tahansa instrumentti antaa tietyn ominaisjoukon vapauksia ja rajoituksia musiikin tekemiseen, myös QBasicin PLAY-lause ja suoraan hertsilukemien määritykseen pohjautuva SOUND-lause ohjaavat muusikon virtuaalista nuottikynää omalla tavallaan. Subjektiivisena kokemuksenani totean, että monofonista ääntä tuottavalla, tekstipohjaisen nuottijonosyntaksin kautta ohjattavalla PLAYllä syntyy aivan erilaisia biisejä kuin vaikkapa pianolla tai kitaralla jammaillen.

Jos pitäisi listata asioita, jotka Redmondin mammutti Microsoft on tehnyt Oikein, luettelon kärkipäässä olisi eittämättä päätös levittää QBasicia MS-DOS-käyttöjärjestelmän oheisohjelmana. Harva asia voi houkutella laajoja kansanjoukkoja koodauksen maailmaan niin hienovaraisen tehokkaasti kuin matalan aloituskynnyksen ohjelmointihiekkalaatikko.

Eräänlainen lippulaivaohjelmisto 1990-luvun BASIC-skenessä oli itse asiassa Microsoftin QuickBASIC. Sillä saattoi kääntää ohjelmat itsenäisiksi .EXE-tiedostoiksi, joiden ajamiseen ei tarvittu tulkkiohjelmaa. QuickBASICin kevytversio QBasic, josta kääntöominaisuus puuttui, oli käytännössä tarjolla joka jampan koneella, kun taas QuickBASIC löytyi kiintolevyn nurkalta paljon harvemmalla koodarilla. Ennen QBasicin voittokulkua MS-DOSin perus-BASIC-tulkki oli niin ikään Microsoftin kehittämä ohjelma, jonka nimi IBM:n PC-koneissa oli BASICA (IBM Advanced BASIC) ja ”IBM-yhteensopivissa” GW-BASIC. Monta muutakin BASIC-versiota on Microsoft vuosien varrella tehnyt: jopa ensimmäinen Microsoftin tuote oli Altair BASIC.

Nicolas Bize lausuu blogissaan QBasicista väitteen, johon on helppo yhtyä. Vaikka QBasicin julkaisemisesta kulunut aika mitataan jo kymmenissä vuosissa, on se yhäkin paras aloittelijan ohjelmointityökalu. QBasicissa kaikki ohjelmointiin tarvittava toiminnallisuus on yhdessä johdonmukaisessa paketissa ilman rasittavia, särkyviä ulkoisia komponentteja ja kirjastoriippuvuuksia. Käyttäjää ei häiritä visuaalisella hälyllä, ja syntaksivirheiden huomaaminen ja koodin ajaminen on tehty helpoksi.

Entä pitääkö arvovaltaisen tietojenkäsittelytieteilijä **Edsger Dijkstran** väite BASICin peruuttamattomasta miel-

Itseironiaa noin vuodelta 1999.

tätuhoavasta vaikutuksesta paikkansa? Kysymystä on pohdittu **Mike Taylorin** blogissa. Mike tulee siihen tulokseen, että Dijkstran näkemys pohjautuu 1970-luvun erittäin kämäiseen BASIC-versioon. BASIC toimii vikoineenkin ihan hyvänä koodikouluna, ja BASICin mahdollisesti päähän pakottamasta spagettikooditekniikasta pystyy oppimaan eroon. Oma mielipiteeni on hyvin samansuuntainen: QBasic tai jokin muu BASIC on varmasti opettanut lukuisille nykypäivänä koodihommia tekeville ihmisille juuri ne oikeat peruskäsitteet kuten silmukat ja ehtolauseet. Ei kauheasti voi valittaa.

QBasicilla tehtyjä ohjelmia

QBasicin mukana levitettiin muutamia virallisia näytepelejä. Eräs niistä on Gorillas, jossa pelaajat ohjaavat kahta heittobanaaneilla taistelevaa gorillaa. Nykyään Gorillas on saatavissa mobiiliversiona. Kun internet (tai siis WWW) yleistyi, viralliset pelit saivat seuraa asianharrastajien alkaessa jakaa hengentuotoksiaan verkossa. QBasic-ohjelmia alkoi ilmestyä ladattaviksi koodarien nettisivuille.

Elävä QBasic-kulttuuri oli hieno asia. Pelejä ja ohjelmia levitettiin tyypillisesti myös levykkeillä esimerkiksi koulukaverien kesken. Harrastajien tekemistä peleistä mieleen on jäänyt muutama erityismaininnan ansaitseva.

Dotroit-nimisessä, värikkäällä ASCII-grafiikalla toteutetussa tappelupelissä hahmot olivat, teknisesti ilmaistuna, *start of heading* (ASCII-arvo 1) ja *start of text* (ASCII-arvo 2). Vaikka ensimmäiset 32 ASCII-arvoa vastaavat tulostumattomia ohjausmerkkejä, näyttötulostuksen koodisivu tarjosi visuaaliset versiot ohjausmerkeistäkin: ASCII-arvot 1 ja 2 näkyivät kahtena erivärisenä hymiönä, jotka sopivat pelin maailmaan parhaalla mahdollisella tavalla.

Toinen on joltain lataussivustolta käsiini saama, **Jason Jacksonin** tekemä, tekstiparserilla ohjattava *Haunted Halloween* -kauhuseikkailupeli. Peli vaikutti pahasti keskeneräiseltä: juoni katkesi kovin lyhyeen, mikä oli suuri harmi, koska pelissä oli potentiaalia. Kenties juuri keskeneräisyydellään *Haunted Halloween* innoitti minua kunnianhimoisimpaan QBasic-peliprojektiini *Slussenin Linna*, johon palaan myöhemmin.

Yhteinen nimittäjä suurimmalle osalle QBasic-pelejä oli se, että pelattavaa per teos oli vähän. Esimerkiksi **Jay Cookin** ja kumppaneiden luomassa roolipelissä nimeltä *Ziel* oli melko vähän sisältöä, vaikka konsoli-RPG-vaikutteiset grafiikat olivat QBasic-mittareilla katsottuna hienot. **Dave Coten** tekemä *Extreme Velocity* -ajopeli edusti hyvää keskitasoa: muutama aika mukavasti suunniteltu kenttä ja kohtalaiset kontrollit. Monet parhaista peleistä olivat sellaisia, joissa yksinkertainen ja toimiva pelimekaniikka mahdollisti pitkät pelisessiot kuten esimerkiksi *Tetris*-klooneissa.

Vuonna 2013(!) tihkui nettiin tietoja **Lance McDonaldin** QBasicilla tekemästä, graafisesti poikkeuksellisen näyttävästä ja kaikesta päätellen pitkähkön keston tarjoavasta *Black Annex* -vakoilupelistä, joka oli hyväksytty Steamiinkin. Tosin kun helmikuussa 2018 tarkistin tilanteen, pelin julkaisupäiväksi Steamissa oli edelleen merkitty ”Coming Soon”…

Kaikki julkaistut QBasic-ohjelmat eivät tietenkään olleet pelejä. Fakiiritempulta tuntui QBasicilla luotu pääteohjelma, jolla saattoi soittaa purkkeihin (BBS). Muistaakseni AT-komennot piti kirjoittaa itse, mutta jos sen osasi tehdä, QBasic-pääte jutteli jollain loitsulla modeemille, ja johan alkoi yhdistämisen kirskuva ääni kaikua kämpässä.

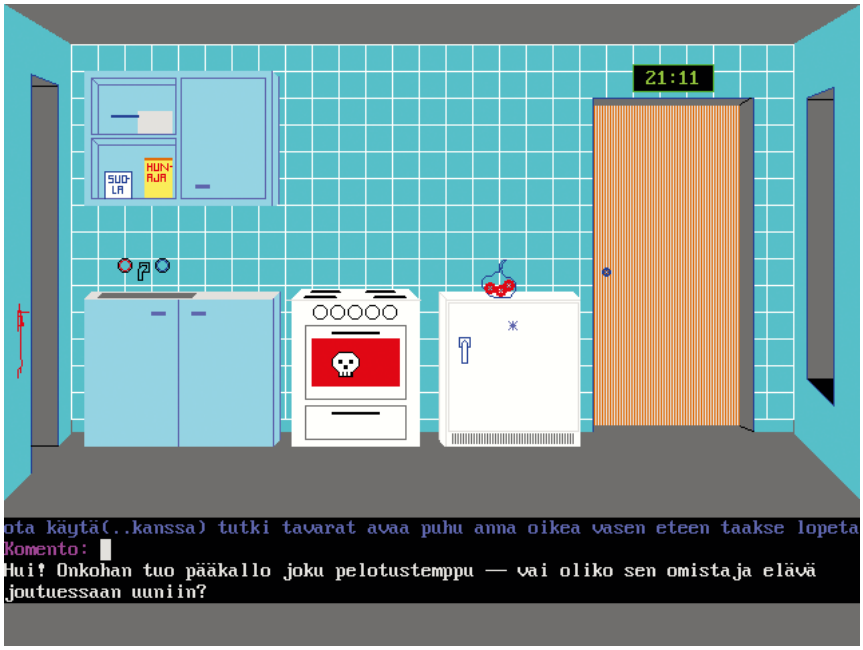
Eräs ohjelmien lajityyppi oli grafiikkademot, joita tehtiin tuohon(kin) aikaan monilla alustoilla, luonnollisesti myös QBasicilla. Muistumia on ainakin jostain ”plasmaefektistä”, joka taisi perustua palettirotaatioon, ja melko hienosta algoritmisesti generoitujen, nousevien tulenliekkien efektistä. Itse tein ”veripisarat valuvat ruudulla” -ohjelman, jota taisin käyttää sitten pelikoodauksessa jonkin hahmon kuolemiskohtauksessa. Vaihtamalla valuvan aineen väriä vaikkapa vihreäksi tai ruskeaksi tehosteen luonnetta saattoi muuttaa kulloinkin haluamaansa suuntaan.

Muistan myös **Ryan**-nimisen amerikkalaisen tyypin tekemän, QBasic-pohjaisen ”käyttöjärjestelmän” *Ryan OS:n* (ROS), jonka slogan taisi olla: ”Better than DOS but requires it.” Lähetin Ryanille jopa sähköpostia ROSiin liittyen, mutta jouduin lainaamaan toisen henkilön sähköpostiosoitetta, koska minulla ei vielä tuolloin ollut omaa. Viestinnässä siis piti huolehtia siitä, että vastaanottaja ei ihmettele lähettäjän ”väärää” nimeä ja toisaalta sähköpostiosoitteen oikealle omistajalle ei aiheudu sekaannuksia.

Vaikka allekirjoittanut on varsin etevä googlaaja, esimerkiksi *Dotroitin* tai *ROS*:n löytäminen nykynetistä on osoittautunut haastavaksi. *Wayback Machine* saattaisi auttaa löytämään edes maininnan näistä ohjelmista. Itse ohjelmatiedostojen löytäminen lienee vielä vaikeampaa. Jollakulla on varmasti tiedostoja tallessa esimerkiksi levykkeillä, mutta niiden pelastaminen vaatii samantapaisia sankarillisia toimenpiteitä kuin harvinaisten C64-pelien konservointi, minkä yksityiskohdista on Skrollissakin kirjoitettu.

Ohjelmointiympäristö

QBasicilla koodista tulee automaattisesti ainakin muotoilujen puolesta melko luettavaa, sillä aina kun uusi rivi on kirjoitettu loppuun, editori



Slussenin Linnan keittiö herättää ruokahalua.

mankeloi rivin koodin yhtenäiseen formaattiin. Ylimääräiset välilyönnit (merkkijonoliteraalien ulkopuolella) katoavat siten, että jäljelle jää yksi välilyönti, ja pienellä kirjoitetut avainsanat (print) muuttuvat isolla kirjoitetuiksi (PRINT). Koodi on helppo ajaa F5-napilla, ja virheilmoitukset ovat melko selkeitä. Ohjelmointikielen ominaisuuksille on editorissa ohjevalikko, josta voi tarkistaa esimerkiksi tietyn sisäänrakennetun apufunktion kutsusyntaksin tai listata kaikki QBasicin tukemat avainsanat. QBasic-kooditiedostojen tiedostopääte on .BAS, joka on oletuspääte monen muunkin BASIC-version tiedostoille.

SUB- tai FUNCTION-syntaksilla voi määrittää aliohjelmia. QBasicin käyttöliittymä kokoaa aliohjelmat näppärästi omaan valikkoon (*View SUBs...*) siten, että niihin voi hypätä ilman tarvetta selata koodia ruudulla 700 rivin päähän. Valikon avulla ohjelmoija pystyy pääohjelmasta siirtymään helposti tiettyyn aliohjelmaan ja takaisin ilman, että hän menettää pääohjelmassa kursorin sijainnin.

Kommenttimerkkinä toimii heittomerkki. Tämä on merkittävä parannus BASICin peruskommenttisyntaksiin, joka alkaa REM-sanalla. Toki myös REM toimii QBasicissa. Huikein yksittäinen syntaksiparannus QBasicissa useisiin muihin BASICeihin verrattuna on kuitenkin rivinumeroiden vapaaehtoisuus. Ohjelmoijan ei siis tarvitse kirjoittaa jokaisen rivin alkuun sitä turhaa, yleensä kymmenen harppauksissa kasvavaa kokonaislukua 10, 20, 30… Tuo parannus oli aikoinaan valtava askel siihen oikeaan suuntaan, että koodatessa voi keskittyä olennaiseen eikä mekaaniseen numeroiden näpyttelyyn. Rivinumeron sijasta tiettyyn kohtaan koodissa voi viitata myös alfanumeerisella tunnisteella (label).

Oletuksena QBasic-ohjelmat toimivat samassa vain tekstitulostuksen sallivassa näyttömoodissa kuin MS-DOS-komentokehote. Tarjolla on myös muutamia erilaisia vapaan piirtämisen mahdollistavia grafiikkatiloja, joita käytettiin tietysti peleissä paljon. Eeppisin grafiikkatila oli mielestäni SCREEN 12, jossa pääsi käsiksi huikeaan VGA-resoluutioon. Värejä sitten oli valikoimassa vain 16 kappaletta, mutta ei määrä ratkaise, vaan se, miten sitä käyttää. Jos oli valmis sietämään isompia pikseleitä, SCREEN 13 -grafiikkatilassa ohjelmoija sai värejä palettiinsa peräti 256 erilaista.

Teknillisiä muisteloita

Kaksi teknistä haastetta olivat sellaisia, että uskoin niiden olevan ratkaistavissa, mutta ratkaisua ei voinut päätellä QBasicin ohjetiedoston perusteella. Ensimmäinen haaste oli äänen soittaminen siten, että yhtäaikaisesti kuva liikkuu ruudulla. Olihan se selvästi mahdollista, koska useimmissa MS-DOS-peleissä musat ja muut äänet rölisivät hienosti samaan aikaan, kun pelihahmot ja viholliset liikkuivat ruudulla. Mutta miten teet saman QBasicilla, kun koodin kaikki muu suoritus pysähtyy piippauksen ajaksi? Entä miten ”rinnakkaisten tapahtumien” todellisuus suhtautuu siihen laajalti toisteltuun väitteeseen, että MS-DOS ei tue moniajoa? Vaistomaisesti kyllä ymmärsin, että moniajosta ei ollut kyse, mutta eipä missään myöskään kerrottu, *mistä* sitten oli oikeasti kyse.

Yksikään helposti saatavilla oleva ohje ei tarjonnut vinkkejä ääniarvoituksen ratkaisuun. Mutta rajoituksethan tulee kääntää voitoksi, joten käytin peleissäni nerokasta suunnitteluparadigmaa: kun animoituun liikkeeseen piti saada viivettä, soitin SOUND-lauseilla jotain lyhyttä törinää animaation framejen välissä. Näin saatiin samalla vaivalla sekä animoituun liikkeeseen kuten kepin heilahdukseen liittyvä ääni että animaation vaatima viive, jota ilman animoitu liike olisi ollut niin nopeaa, ettei sitä olisi ehtinyt edes nähdä. Yksi frame siis näkyi juuri niin kauan kuin kaiken muun toiminnan kanssa ei-rinnakkainen piipperinsoitto pakotti ruudun näkymän paikoilleen jäätyneeksi.

Animaatiotekniikka peleissäni oli muutenkin äärimmäisen tehokasta: homma perustui siihen, että edellisen framen jälkeen piirrettiin BF-optiota (”box, fill”) käyttävällä LINE-komennolla musta peittävä muoto juuri oikeaan kohtaan ruutua ja sitten luotiin uusi frame näin syntyneen tuoreen piirtopinnan päälle. Animoitujen kohteiden taustat tuppasivat sitten jostain syystä olemaan peleissäni yksivärisen mustia.

Toinen jännistä kysymyksistä koski kuvia. Selväähän oli, että peligrafiikat tyypillisesti ladataan jonkinlaisista datatiedostoista, eikä niitä useinkaan piirretä käyttäen geometrisia ympyröitä ja viivaprimitiivejä (QBasicin CIRCLE ja LINE). Miten kuvatiedostotuen saisi mukaan pelimoottoriin?

Joku guru oli tehnyt QBasic-ohjelman, jolla BMP-tiedostoista saattoi ladata kuvia ruudulle grafiikkatilassa. Äkkiseltään voisi luulla, että tämän koodin käyttäminen omissa ohjelmissa olisi ratkaissut kuvatiedosto-ongelman. Tosiasiassa kaksi merkittävää jatko-ongelmaa tekivät tästäkin helpommin sanotun kuin tehdyn ominaisuuden.

Ensinnäkin kuvat latautuivat suo-

raan ruudulle, eivät jonkinlaiseen puskuriin, josta niitä olisi voinut toistuvasti piirtää ruudulle haluttuihin kohtiin. Toisekseen kuvan latautuminen kesti kuin nälkävuosi: kun ihmeohjelma luki dataa tiedostosta, ruudulle ilmestyvien pikselien vana eteni niin hitaasti, että sitä saattoi seurata silmällä. BMP-tiedoston lukeminen ei ollut ainoa merkittävä hitauden lähde. Tein hieman tutkimuksia, joissa paljastui kunnon pommi: ylipäätään pikselien piirto ruudulle FOR-silmukassa PSET-lauseella oli jäätävän hidasta, vaikka silmukassa ei olisi mitään muuta suoritettavaa kuin koordinaattimuuttujien päivitys ja PSET!

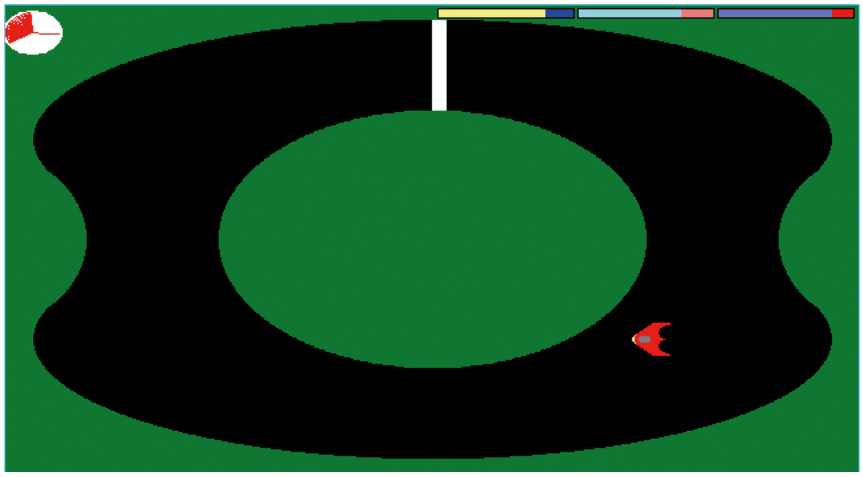
Kuinka koskaan voisi olla mahdollista näyttää nopeasti mutta pikselipohjaisen datan avulla minkäänlaisia kuvia, joiden pinta-ala olisi puolikasta postimerkkiä isompi? Muistaakseni hyödynsin kuvatiedostoja vain yhdessä pelinkyhäelmässä, jossa oli Doom-tyyliin asteittain energian vähentyessä ruhjotummaksi muuttuva naama ruudun reunassa.

Myöhemmin kokeilin BSAVE- ja BLOAD-pohjaista kuvien latausta, joka toimi nopeammin. Kaikenlaista jännää saattoi tehdä suoraan muistia käsittelevillä toiminnoilla kuten DEF SEG, PEEK ja POKE, mutta nämähän ovat malliesimerkki siitä osastosta, jonka hyötykäyttö ei onnistu pelkän QBasic-ohjetekstin pohjalta.

Direkt från Slussen

Toinen kahdesta parhaasta tekemästäni QBasic-pelistä on *Slussen Fighters*. Nimen Slussen-sana tulee peruskoulun *Linje*-ruotsinkirjasarjasta, jossa mainittu Tukholman Slussen alkoi elää aivan omaa elämäänsä minun ja parin kaverini mielikuvituksessa. Ruotsi liittyi QBasiciin toistakin kautta: luokkaretkirahastoa varten koodasin ruotsin opetusohjelman RUOTSI.BAS, jota myytiin levykkeellä luonnollisestikin RUOTSI.EXE-ohjelmaksi käännettynä. Hinta oli 30 markkaa.

Alun perin *Slussen Fightersin* nimeksi oli tarkoitus tulla *Slussement Fighters*. Nimen ment-lopukkeen taustalla oli konsolipeli *Teenage Mutant Ninja Turtles: Tournament Fighters*. Vaikutteita otin *Street Fighterin* ja *Mortal Kombatin* kaltaisista tappelupeleistä. *Slussen Fightersin* henkinen edeltäjä (jos muistan tiedostonimen oikein) oli



Extreme Velocityn aurinkokunta-ajokisojen Maa-planeetan kenttä.

TAPPELU.BAS, joka oli paljon yksinkertaisempi tikku-ukkojen tappelupeli. Hengaillessani Nokialla TET-harjoittelussa kokeilin TAPPELU.BASia yhdellä läppärillä. Havaitsin, että läppärin audiovisuaaliset ominaisuudet olivat rajoittuneita: äänet eivät kuuluneet, ja keltainen väri oli kyseisen koneen ysäri-LCD-ruudulla paleltuneen palsternakan sävyistä.

Eräs genre, jonka pelien koodausta tuli harrastettua parhaassa BASIC-hengessä, oli tietenkin tekstiseikkailut. Graafinen pelinäkymä yhdistettynä tekstipohjaiseen komentoparseriin tuntui kuitenkin vetoavimmalta ilmaisumuodolta, ja näin syntyi *Slussenin Linna*, jossa ensimmäisestä persoonasta synkeän arvoituslinnan huoneita tutkiva pelaaja yrittää pelastaa kidnapatun kaverinsa nimeltä Eino.

Aiemmin mainitun *Haunted Halloweenin* lisäksi voimakkaita vaikutteita *Slussenin Linnan* pelisuunnitteluun antoi LucasArtsin seikkailupelien maailma, erityisesti kaikkien aikojen suosikkini *Day of the Tentacle*. LucasArtsin SCUMM-pelimoottorin fiksusti suunniteltu, muutamiin yleiskäyttöisiin toimintoverbeihin perustuva logiikka näkyy *Slussenin Linnan* kontrolleissa, tyyliin: ”ota kuorimaveitsi”, ”käytä kuorimaveistä perunan kanssa”. (Nämä eivät olleet spoilereita. Pelissä ei ole oikeasti kuorimaveistä.) *Slussenin Linnassa* toimintoverbit näkyvät SCUMM-tyylisesti ruudulla koko ajan, joten niitä ei tarvitse muistaa ulkoa tai kaivaa kerta toisensa jälkeen jostain manuaalista. Pelin suomenkielisyys saattoi vaikuttaa kansainvälisen läpimurron tapahtumatta jäämiseen. *Slussen Fightersista* tein taas versiot sekä suomeksi että englanniksi.

Uusi QBasic?

QBasicin kaltaiselle helposti lähestyttävälle ohjelmointiympäristölle on tarvetta nykypäivänäkin. En ole itse perehtynyt esimerkiksi melko tunnettuun Scratchiin, jonka suunnittelijoiden kunnioitettavana päämääränä on ohjelmoinnin opettaminen uudelle sukupolvelle.

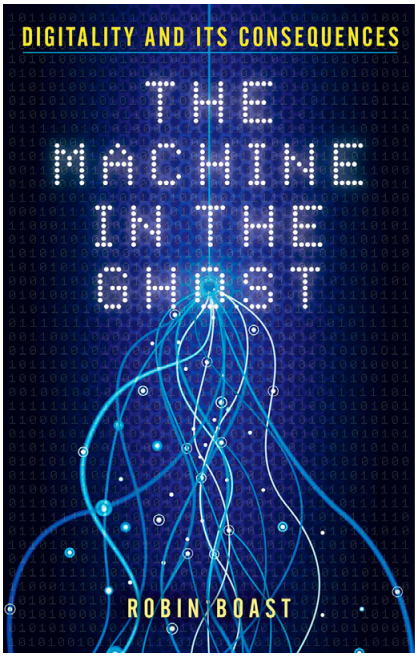
Eräs syy QBasicin suosioon aikoinaan oli sen laaja saatavuus aikakautensa PC-tietokoneilla ilman erillistä asennusvaihetta. Esimerkki nykymaailmassa tuon vaatimuksen täyttävästä ohjelmointikielestä voisi olla JavaScript: jokaisessa tavallisessa nettiselaimessa on JavaScript-tuki, joten periaatteessa koodaukseen tutustumisen voi aloittaa vaikkapa Chromen tai Firefoxin (ja tekstieditoriohjelman) parissa. Tokikaan JS ei ole niin aloittelijaystävällinen kokonaisuus kuin QBasic aikoinaan.

Nykykoneilla QBasicia ei voi ajaa helposti, vaan väliin tarvitaan DOSBoxin kaltaisia erikoisohjelmia. QB64-niminen harrastajien luoma ohjelmisto mahdollistaa QBasic-ohjelmien ajamisen muun muassa uusissa Windows-käyttöjärjestelmissä. Microsoft on 2000-luvulla julkaissut Small Basic -nimisen BASIC-version. Pienuutta lupailevasta nimestään huolimatta Small Basicin ajamiseen tarvitaan .NET-framework, mutta ainakin kielen syntaksi on ilahduttavan minimaalinen. Tavoite eli uusien ihmisten houkuttelu koodauksen maailmaan on Small Basicillakin hieno, QBasicin kaltaista legendaa siitä ei vain taida tulla.

The Machine in the Ghost – Digitality and its Consequences
Digitaalisuuden syvin olemus

Amsterdamin yliopiston kulttuurisen tietojenkäsittelytieteen professori, **Robin Boast***, on kirjoittanut kirjan, joka valaisee digitaalisuuden historiaa ja sen luonnetta.*

Teksti: Jari Sihvola



Elämä yhteiskunnassamme on nivoutunut yhteen digitaalisuudeksi kutsutun ilmiön kanssa jo vuosikymmenien ajan, mutta mikä on digitalisaation varsinainen olemus ja sen rooli teknologian kehityksessä?

Boastin viitoittama matka digitaalisuuteen alkaa 1870-luvulta **Émile Baudot'n** luoman kaukokirjoittimen koodista, jonka jäänteitä löytyy yhä modernien tietokoneiden isojen kirjainten enkoodauksesta. Baudot päätyi digitaaliseen koodiin, koska hän halusi käyttää tiedonsiirron luotettavuuden varmistamiseksi kaikista yksinkertaisinta kuviteltavissa olevaa enkoodausta. Binäärisiä koodijärjestelmiä oli nähty aiemminkin, mutta Baudot'n kirjoittimen binäärikoodi oli pituudeltaan vakiopitkää (jokainen merkki viisi bittiä), kuten moderni digitaalikoodaus.

Myös Morsen aakkosia pidetään usein digitaalisina, mutta Boast kiistää tämän, koska hänen mielestään koodin yksikön vakiopituus on digitaalisuuden edellytys. Kaukokirjoittimiin perustuva digitaalinen viestintä yleistyi ensimmäisen maailmansodan jälkeen. Näin ollen 1950-luvulla ensimmäisiin digitaalitietokoneisiin oli jo valmiiksi olemassa teknologiat digitaalisen datan syöttämiseen ja tulostamiseen.

Boastin kertomassa historiassa keskeistä on, etteivät digitaaliset tietokoneet olleet mikään itsestäänselvyys, vaan niiden hyödyt opittiin tuntemaan vähitellen ja pienin askelin. **Alan Turingin** hahmottelema myöhempiä tietokoneita ennakoinut teoreettinen malli, *universaali kone*, ei horjuttanut analogisuuden asemaa. Turing kyllä osoitti, että tietokone voisi toimia binäärilaskennan keinoin, muttei väittänyt, että sen pitäisi olla digitaalinen tai binäärinen.

Toisen maailmansodan jälkeen digitalisuuden suuri innovaattori, **Claude Shannon**, oli jo keksinyt algoritmein toimivat piirilevyt, joissa releet toimivat loogisina operaattoreina. Shannon oli myös osoittanut teoreettisesti digitaalisen enkoodauksen olevan tehokkuuden kannalta paras ratkaisu tiedonsiirrossa, sekä sen, että mitä tahansa tietoa voidaan tallentaa digitaalisessa muodossa. Näistä suurista löydöksistä viimeinen sinetöi digitaalisuuden ja median tiiviin liiton. Mutta kuten Boast muistuttaa, Shannonkaan ei vielä ehdottanut digitaalisen tietokoneen rakentamista.

1940-luvun teknologialla analoginen tietokone oli digitaalista nopeampi, ja analogisten valta-asema jatkui vielä 1950-luvun ajan. Analogisissa tietokoneissa esimerkiksi pluslasku tapahtuu lisäämällä suoraan yhteen kaksi eri suuruista jännitettä, joiden koot ovat analogisia suhteessa laskettaviin lukuihin. Digitaalitietokoneessa kaikki on monimutkaisempaa: datan binäärinen linjakoodi ei sisällä mitään analogiaa eikä edes suoraan viittaa binäärilukuihin, vaan se joudutaan ensin muuntamaan binääriluvuiksi. Pelkkä yksinkertainen pluslasku edellyttää ainakin neljäntoista loogisen portin (esim. transistorien) käyttöä.

Digitaalisten tietokoneiden kykyä käsitellä loogisia algoritmeja kuitenkin tarvittiin, ensimmäisenä *Whirlwind 1* -tietokoneen (1948) syntyyn johtaneessa Yhdysvaltain armeijan lentosimulaattorihankkeessa. Digitietokoneen haasteena olleet teknologiset esteet saatiin raivattua keksimällä ferriittirengasmuisti sekä rinnakkaisprosessointi, ja 1950-luvun lopulla alkoivat yliopistot Euroopassa ja Yhdysvalloissa perustaa omia digitietokoneiden hankkeita. Samoin 1950-luvulla ymmärrettiin Yhdysvaltain armeijan *Fieldata*-järjestelmän käytön myötä digitaalilaitteiden mahdollisuudet integroida dataa eri lähteistä siinä missä analogilaitteet käsittelivät vain yhden laitteen dataa.

Digitaalisen tietokoneen kehityksen historiassa Boast painottaa **Doug Engelbartin**, **Ted Nelsonin** ja **Alan Kayn** kaltaisten uskaliaiden visionäärien roolia. Heidän ajattelunsa painotti tietoverkkoihin kytkettyjen henkilökohtaisten tietokoneiden merkitystä 1960- ja 1970-luvulla jo selvästi ennen henkilökohtaisten koneiden aikakautta. Engelbart hahmotteli 1960-luvun alussa hieman Google Docsia muistuttavan vision yhteistyöjärjestelmästä, johon sisältyi jo modernin henkilökohtaisen tietokoneen kaikki olennaiset piirteet hiirtä myöten.

Xerox Parcissa työskennellyt Kay kehitti uutta lapsille ja opiskelijoille suunnattua tietokoneistettua oppimisjärjestelmää. Tämä työ ohjasi häntä graafisen käyttöliittymän ja henkilökohtaisten koneiden suuntaan. Nelsonin visio tietoverkosta yksinkertaisella käyttöliittymällä synnytti hypertekstin ja virtuaalisuuden käsitteet ennakoiden nykyistä internetiä. Vaikka näiden visionäärien omat it-hankkeet eivät menestyneetkään, he onnistuivat inspiroimaan muita laitekehittäjiä.

Boast näkee, että digitaalisuuden olemus oli tietoteknisten innovaatioiden myötä saavuttanut olennaisen muotonsa 1980-luvun puoliväliin tultaessa, jolloin graafiset käyttöliittymät oli otettu käyttöön. Vaikka kehitys jatkoi toki sen jälkeen kehittymistään, oli tuolloin jo hahmottunut, mitä digitaalisuus on ja mihin sitä käytetään. Laskimesta oli kehittynyt muutaman vuosikymmenen kuluessa mediaa käsittelevä kone, ja juuri tämä kyky kä-

sitellä monipuolisesti mediaa onkin Boastin ajattelussa digitaalisuuden olemuksen ydin.

Boast vastustaa mielikuvaa digitaalisesta datasta stabiilina tietovarantona, joka mekaanisesti renderoidaan käyttäjän näkyville. Hän painottaa sen sijaan digitaalisessa tiedonkäsittelyssä tapahtuvaa jatkuvaa objektien kopiointia ja muokkaamista. Siinä missä analoginen media toistetaan lähes sellaisenaan ehkä pienten säätöjen lävitse, digiteknologiassa itse mediaobjektia muokataan sekä normaalikäytössä esityksen aikana että aina, kun sitä siirretään eri konteksteihin.

Boast vertaa digitaaliteknologiaa jopa teatteriin siinä mielessä, että molemmissa luodaan esitys aktiivisesti joka kerta hieman eri tavalla. Myös mahdollisimman samanlaisena toistetut digitaaliset prosessit (”algoritmiset performanssit”) käyttävät aina eri muistipaikkoja ja prosessointiportteja. Tätä prosessia Boast kuvailee ”luovaksi” ja painottaa, että digimedian käyttäjä osallistuu itsekin tähän algoritmeilla toteutettavaan prosessiin, eikä ainoastaan sen lopputulokseen.

Koska digitaalisuutta luonnehtivat jatkuva kopiointi ja muokkaus, Boastin luonnehdinnan mukaan alkuperäinen mediaobjekti tarkoituksineen menettää digitaalisessa mediassa merkityksensä. Toiseksi digitaalisuutta määrittäväksi piirteeksi hän nostaa interaktiivisuuden, joka luonnehtii digitaaliajalla kehitettyjä laitteita. Käyttäjän passiivisemmaksi jättävät digilaitteet, kuten CD-soitin, ovat hänestä vain vanhemman teknologian digitaalisia uudelleentoteutuksia.

Vaikka digitaalisuuden voittokulku näyttää ehkä selvältä, Boast tuo yllättäen esille myös, että nykyään tutkitaan yhä myös analogisten tietokoneiden mahdollisuuksia. Niiden on uskottu olevan digitaalisia serkkujaan nopeampia ainakin hyvin suuria datamääriä sekä Fourier’n muunnoksia yhdistävissä laskentatehtävissä. Tulevaisuudesta emme voi tietää, mutta Boastin kirja on mielenkiintoinen ja opettava kertomus siitä, mihin asti olemme digitaalista polkua pitkin tähän mennessä saapuneet.

Robin Boast, *The Machine in the Ghost – Digitality and its Consequences*
Reaktion books, London, 2017, 195 s.

Disobey
Selviytymisopas vuoden hakkeritapahtumaan

Jos olet kiinnostunut tietoturvasta tai hakkeroinnista, tule ensi vuonna Disobeyhin. Jos haluat osallistua hakkerointihaasteeseen, ota kavereita mukaan.

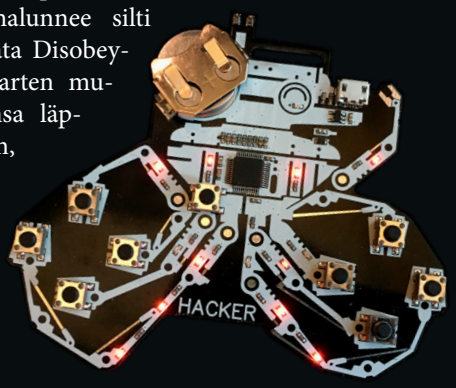
Teksti: Valhe Kouneli, Tapio Berschewsky
Kuvat: Disobey, Nasu Viljanmaa

Disobey on paras hakkeritapahtuma Suomessa, riippumatta lähtötasostasi. Kerroimme tapahtumasta ja sen tavoitteista laajasti viime vuonna (Skrolli 2017.1); tieto on yhä ajankohtaista. Tällä kertaa pureudumme siihen, miten harrastuneemmalle kohdeyleisölle tarkoitettuun päähaasteeseen kannattaa varautua. Disobeyn runsaaseen muuhun sisältöön voit tutustua lukemalla oheisen kainalojutun signaalihakkeroinnista ja tsekkaamalla esitykset tapahtuman Youtube-kanavalta.

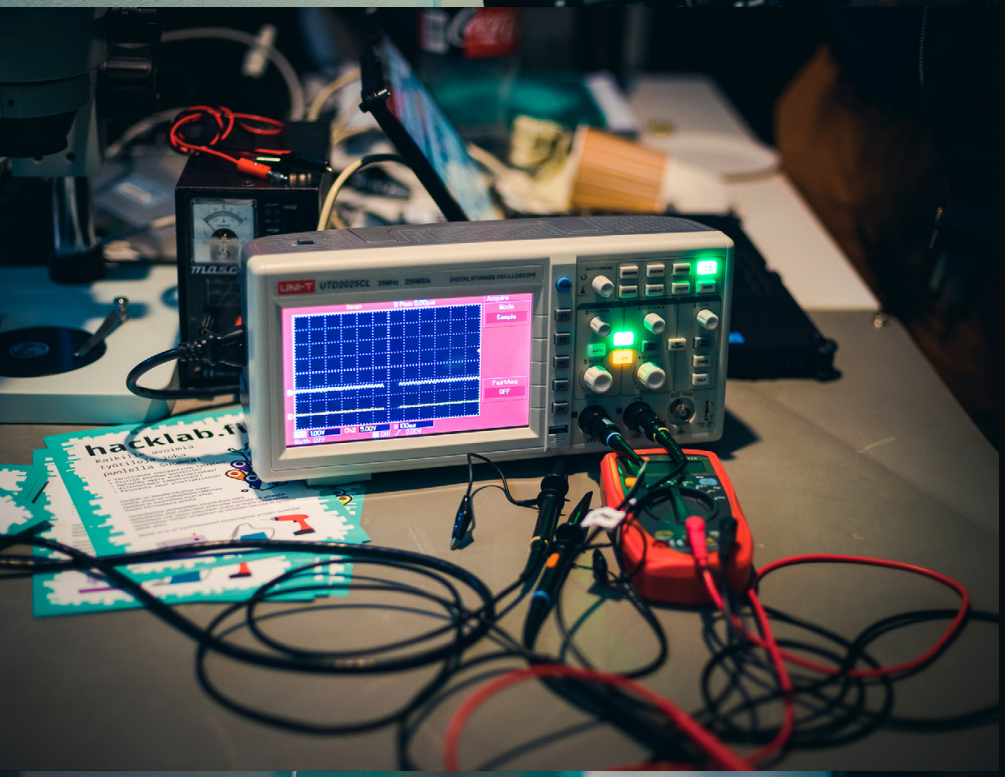
Disobeyn jokavuotinen haaste koostuu erilaisista piilotetuista vihjeistä, jotka johtavat seuraaviin vihjeisiin, jotka taas seuraaviin ja niin edelleen. Pelissä eteenpäin päästäkseen on osattava tunnistaa vihjeet ja tulkittava ne oikein. Monipuolinen tietämys erilaisista hakkerointiin kytkeytyvistä asioista kuten elektroniikasta ja tietoliikenneprotokollista on tarpeen, mutta myös loogista päättelykykyä ja luovuutta vaaditaan. Mukaan kannattaa koota noin neljän hengen tiimi ja varustautua hyvin. Haasteesta saa eniten irti joukolla.

Osassa vinkeistä vaaditaan myös nörttikulttuurin yleistuntemusta. Esimerkiksi tämän vuoden haasteessa piti muun muassa tietää, mikä on Konamikoodi, sekä tunnistaa Commander Keen -fontti, japanilaiset numeromerkit ja Star Warsissa esiintynyt Viper probe droid. Etenkin tässä asiassa useampi apupää tulkitsemassa vihjeitä on hyödyksi.

Haasteessa ei päässyt etenemään loputtomasti ottamatta yhteyttä Disobeyn omaan, niin kutsuttuun hostiilin paikallisverkkoon. Kuten viime vuonna, tänäkin vuonna se tarkoitti koneensa antamista vapaaehtoisesti kaikenlaisten hakkerointiyritysten kohteeksi. Toisaalta, kuten allekirjoittanutta lohdutettiin, ”hakkerit eivät pasko omalle tontilleen”. Vaikka lupa oli rellestää, tiedä sitten kuinka moni niin oikeasti teki. Tietoturvastaan epävarmempi tai tarkempi halunnee silti pakata Disobeytä varten mukaansa läppärin,



Kävijäbadget olivat tänäkin vuonna pieniä, elektronisia laitteita. Laitteen toiminta, piirilevylle painetut kuvat ja kuvakoodit sekä badgen nauhasta löytyvät merkit toimivat ensimmäisinä hakkerointihaasteen vihjeinä. Badgen väri ja toiminnallisuus muuttui hieman sen mukaan, minkä tyyppinen kävijä oli kyseessä, ja haasteeseen osallistuneiden piti sekä kommunikoida keskenään että laittaa badgensa kommunikoimaan keskenään päästäkseen haasteessa eteenpäin. Sellaisenaan badge toimii myös peliohjaimena, joten tapahtuman jälkeen kotiinvietäväksi ei jäänyt pelkkää elektroniikkaromua.



joka ei sisällä sensitiivistä dataa, mutta jolle on kuitenkin asennettu valmiiksi erilaisia hyötyohjelmia. Arkikäyttöläppärinsä tapahtumaan mukaan ottavan kannattaa vähintään varmistaa, että uusimmat tietoturvapäivitykset on ajettu.

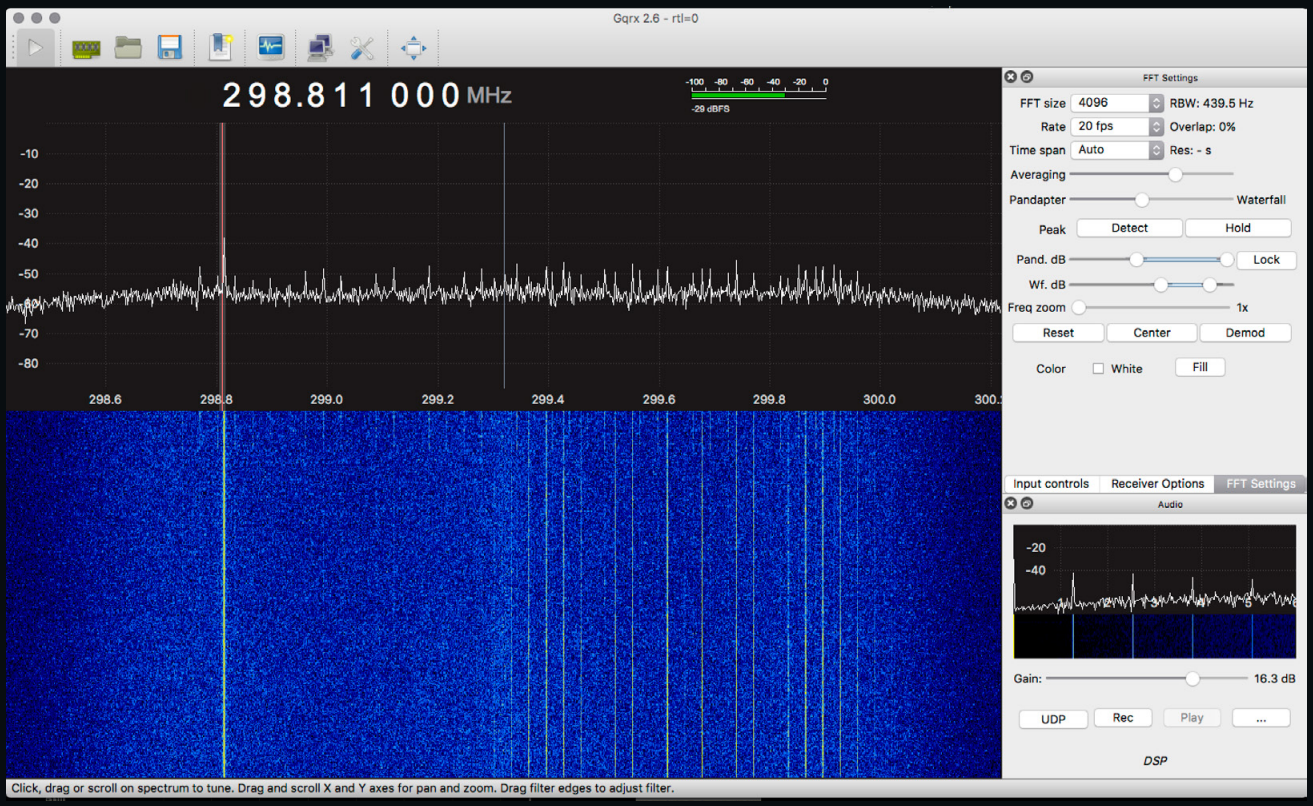
Lisää pakkaamisvinkkejä ensi vuotta varten: Omat wlan-verkot oli kielletty Disobeyssä, joten mukana on hyvä olla oman älypuhelimen ja läppärin välille sopiva datapiuha. Myöhemmissä haasteen vaiheissa ja eri tapahtumasponsorien omissa haasteissa saattoi hyötyä spesifeistä työkaluista, kuten esimerkiksi steganografiasoftasta, heksaeditorista, sopivasta ohjelmointiympäristöstä tai virtuaalikoneesta. Sellaisia on hyvä olla ”pakattuna mukaan” omalle läppärille valmiina. Porttiskanneria tarvitsee tuskin mainita. Tänä vuonna hyötyä oli myös micro-usb-piuhasta, mutta sellaisen olisi varmaan saanut tarvittaessa lainaan joltakulta.

Kukaan ei ratkaissut vuonna 2018 Disobeyn virallista hakkerointihaastetta loppuun asti. Moni pääsi kuitenkin pitkälle, ja tapahtumasponsoreiden haasteista moni ratkaistiin. Aiheesta kiinnostuneiden kannattaa pysyä kuulolla, sillä saatamme julkaista blogissamme (skrolli.fi/blogi) tai seuraavassa numerossa vielä lisää tietoa Disobeyn 2018 eri haasteista ja niiden ratkaisuista.

Disobeyn keskiössä ovat haasteen lisäksi esitykset, työpajat ja uusien kontaktien luominen. Yksi työpajoista oli niin haluttu, että osallistujalista täyttyi hetkessä. Pyysimme siis **Oona Räisästä** kirjoittamaan sen missanneille oheisen käsittelyn signaalihakkerointityöpajan sisällöstä.

Mikä Disobey?

Disobey on nyt kolme kertaa järjestetty tapahtuma, jonka tarkoituksena on levittää muillekin sitä tietotaitoa, joka hakkereilla on. Ajatuksena on edistää ihmisten kykyä suojautua alati vaaralliselemmassa digitaalisessa ympäristössä niiltä uhilta, joiden taltuttamiseen valtiot tai yritykset eivät pysty. Disobeyn pääjärjestäjä on Skrollinkin avustajalistoilta löytyvä **Benjamin ”NotMyNick” Särkkä**. Lisätietoa tapahtumasta löytää osoitteesta disobey.fi tai liittymällä ircnetin #disobey.fi-kanavalle.



Gqrx-ohjelma näyttää RTL-SDR-tikun vastaanottamaa signaalia reaaliajassa.

Signaalien salat

Oona Räisänen

Signaalien tutkiminen on ollut harrastukseni jo pitkään. Pidin Disobeyssä ilahduttavan monia kiinnostaneen työpajan signaalien perusasioista ja erityisesti harrastuksessa käyttämistäni työkaluista.

Signaaleilla tarkoitan mitä tahansa mitattavissa olevaa asiaa, joka muuttuu ajan suhteen: usein se on radioaalto tai ääni, mutta se voi olla myös sähkönkulutus, valon määrä tai vaikka GPS-sijainti.

Yksinkertaisimmillaan tällainen mittaustyökalu on vaikkapa iPhone-sovellus SpectrumView, joka näyttää puhelimen mikrofoniin tulevan äänen taajuusjakauman reaaliaikaisesti. Sovellusta kokeilemalla saa hieman käsitystä siitä, miten ihmiskorvalle tutut aistimukset kuten äänenväri ja äänenkorkeus ovat kytköksissä signaalien taajuuksiin ja taajuusjakaumaan. Älypuhelin myös kuulee jonkin verran korkeampia ääniä kuin ihminen, joten käytän tätä sovellusta joskus ultraäänien lähteiden etsimiseen.

Radiosignaalien vastaanottamiseen tarvitaan hieman lisälaitteita. Halvimmillaan se onnistuu tietokoneen USB-porttiin kytkettävällä, parinkymmenen euron hintaisella RTL-SDR-radiolla. Nämä olivat alun perin digi-tv-vastaanottimiksi kehitettyjä laitteita, joita hakkerit oppivat virittämään alkuperäisten taajuusalueiden ulkopuolelle. Nyt niillä voi kuulla kaikenlaista yleisradiolähetyksistä radioamatöörikeskusteluihin. Radiosignaaleja kuunnellessa on kuitenkin muistettava, että lain mukaan niiden sisältöä ei saa mitenkään välittää eteenpäin, elleivät ne ole julkisiksi tarkoitettuja.

Tällaisen USB-tikkuradion saa toimimaan vaikkapa Gqrx-ohjelman avulla. Ohjelmalla voi virittää tikun haluamalleen taajuudelle, tarkastella näkyykö taajuuden lähistöllä lähetyksiä ja myös kuunnella analogisia AM- ja FM-radiolähetteitä. Tällaista tietokoneavusteista radioviestintää kutsutaan yleisemmin ohjelmistoradioksi eli SDR:ksi.

Viime aikoina olen ollut enemmän kiinnostunut digitaalisista radiolähetteistä. On kiinnostava palapeli tallentaa pätkä lähetystä ja pohtia, millä tavoin data siirtyy radioaallon mukana. Onko bitit koodattu aallon amplitudin vai taajuuden vaihteluun? Vai voisiko mukana olla jotain ovelampaa, kuten vaihesiirtymiä? Kuinka nopeasti dataa siirtyy ja kauanko yksi bitti kestää? Muodostavatko bitit tavuja tai muunlaisia ryhmiä?

Tallennettujen signaalien tutkimiseen käytän *baudline*-ohjelmaa. Se tarjoaa monenlaisia havainnollistavia näkymiä ja mittatyökaluja. Esimerkiksi bittinopeuksien tai muun rytmisyyden määrittämiseen voi käyttää erityistä aikajaksokursoria. *Baudline* on ilmainen, mutta sen lähdekoodi on suljettu. Avoimen lähdekoodin ohjelmistoista esimerkiksi *Inspectrum* on samankaltainen.

Komentorivityökaluista *sox* (sound exchange) on ehdoton suosikkini. Sillä voi esimerkiksi helposti muuntaa signaalin näytteistystaajuuden tai purkaa äänitiedostoja helposti käsiteltäväksi raa'aksi PCM-dataksi. *Minimodem*-ohjelmalla voi purkaa hyvin yleistä taajuusavainnusta eli FSK:ta käyttäviä digitaalisia lähetteitä. *Multimon-ng* puolestaan purkaa useita muita lähetteitä, kuten lankapuhelimien DTMF:ää ja hakulaitteiden POCSAGia.

Signal Identification Wiki ja Viestintäviraston taajuusjakotaulukko ovat hyviä internet-lähteitä signaalien alkuperän selvittämiseen. Monille lähetteille on myös yleensä olemassa jokin standardi tai kuvausdokumentti, jonka avulla sisältö voidaan selvittää.

Pähkinänurkka
Neliöjuuritehtävän jälkipyykki

Skrollin toimituksen blogin helmikuun ohjelmointipähkinä saavutti valtavan suosion. Ongelmana oli jakaa luvut √1, √2, …, √100 kahteen joukkoon siten, että osien summat ovat mahdollisimman lähellä toisiaan.

Jarno Niklas Alanko

Ongelman esitti tiedettävästi alun perin tietojenkäsittelytieteilijä **Bob Floyd** vuonna 1972 joukolle ensimmäinen vuoden tohtoriopiskelijoita Stanfordissa. Alkuperäinen muotoilu oli:

”The numbers √1, √2, … , √50 are to be partitioned into two parts whose sum is nearly equal; find the best such partition you can, using less than 10 seconds of computer time.”

Tällaisenaan ongelma ei sovi nykypäivään, koska prosessoreiden laskentateho on kasvanut niin paljon, että tämä ongelma on ratkottavissa täsmällisesti. Siispä blogia varten tuplasimme syötteen koon, joka kasvattaa hakuavaruuden taas mahdottoman suureksi.

Jos haluat miettiä ongelmaa ensin itse, lopeta lukeminen tähän!

Pulmaa kannattaa lähestyä muotoilemalle se uudelleen seuraavasti: etsi osajoukko, jonka summa on mahdollisimman lähellä kaikkien neliöjuurten summan puolikasta, joka on noin 335,7. Nyt ongelma muistuttaa tunnettua *subset sum* -ongelmaa, jossa on etsittävä annetusta lukujoukosta osajoukko, jonka summa on annettu luku. Subset sum on NP-vaikea ongelma, joten tehokasta algoritmia tuskin löytyy. Ongelma on kuitenkin hyvä harjoitus suurten hakuavaruuksien heuristiseen läpikäyntiin.

Donald Knuth kirjoittaa ongelmasta artikkelissaan ”Are Toy Problems Useful?” vuodelta 1977. Artikkeli löytyy tänä päivänä helpoiten Knuthin kirjasta *Selected Papers on Fun and Games*. Knuthin mielestä ongelma on täydellinen leluongelma: se on tarpeeksi yksinkertainen, että siinä ei juutu merkityksettömiin yksityiskohtiin, mutta siinä on silti tarpeeksi mielenkiintoista rakennetta, kuten esimerkiksi se, että √3 + √12 = √27.

Knuthin ratkaisu toimii pääpiirteittäin seuraavasti: otetaan sivuun ne luvut, joiden neliöjuuri on kokonaisluku: 1, 2, 4, 9, 16… Sen jälkeen etsitään jäljellä olevista luvuista osajoukko, jonka summan *desimaaliosa* on mahdollisimman lähellä kaikkien neliöjuurten summan puolikkaan desimaaliosaa. Lopuksi yritetään korjata kokonaislukuosa kohdalleen ottamalla sivuun laitetuista luvuista sopiva osajoukko mukaan. Knuthilla on myös käytössä arsenaali muita huomioita kuten se, että √8 = 2√2, √18 = 3√2, √32 = 4√2, ja niin edelleen. Yksityiskohdat on kirjoitettu hienosti auki Knuthin artikkelissa, joten emme käy niitä läpi tässä.

Lukijoiden ratkaisut

Toimitus seurasi jännityksellä, kun kommentteihin ilmestyi toinen toistaan parempia ratkaisuja. Ensin päästiin 32-bittisen floating point luvun tarkkuuden rajoille, sitten 64-bittisen, ja lopulta vielä pitemmälle. Parhaan ratkaisun löysi lopulta nimimerkki **sisu** tuloksella 3,54e-17, kuusi kertaluokkaa muita edellä! Sisun ratkaisu on seuraava ositus:

[1, 2, 3, 5, 6, 10, 11, 14, 15, 18, 22, 26, 28, 32, 33, 36, 37, 39, 41, 42, 44, 45, 46, 49, 50, 51, 54, 55, 62, 63, 64, 65, 67, 68, 69, 70, 73, 75, 78, 79, 80, 82, 85, 88, 89, 92, 93, 96, 97, 99]

[4, 7, 8, 9, 12, 13, 16, 17, 19, 20, 21, 23, 24, 25, 27, 29, 30, 31, 34, 35, 38, 40, 43, 47, 48, 52, 53, 56, 57, 58, 59, 60, 61, 66, 71, 72, 74, 76, 77, 81, 83, 84, 86, 87, 90, 91, 94, 95, 98, 100]

Miten tähän oli päädytty? Hakuavaruuden koko on 2^{100}, joten kaikkia vaihtoehtoja ei voi millään käydä läpi. Sen sijaan on keksittävä algoritmi, joka keskittyy jollain tapaa hakuavaruuden lupaavimpiin osiin. Tähän on olemassa tunnettuja metaheuristiikkoja kuten simuloitu jäähdyttäminen ja geneettiset algoritmit.

Sisun ratkaisu ei kuitenkaan käytä näitä, vaan se on yhdistelmä arpomista ja täsmällistä ratkontaa. Ratkaisu toimii kahdessa vaiheessa: ensin se arpoo jonkin osajoukon luvuista √51, …, √100 ja sitten täydentää osajoukkoa valitsemalla luvuista √1, …, √50 osajoukon siten, että arvotun ja valitun osajoukon summa on mahdollisimman lähellä kaikkien neliöjuurten summan puolikasta.

Sisu ratkaisee toisen osuuden täsmällisesti käyttämällä *meet-in-the-middle*-tekniikkaa. Se toimii seuraavasti: luodaan kaksi taulukkoa A ja B. Lasketaan taulukkoon A kaikki summat, jotka on mahdollista muodostaa luvuista √1, … , √25, ja taulukkoon B kaikki summat, jotka voidaan muodostaa luvuista √26, …, √50. Järjestetään molemmat taulukot suuruusjärjestykseen. Nyt voidaan etsiä paras kokonaissumma käymällä läpi taulukko A alusta loppuun ja taulukko B lopusta alkuun samanaikaisesti.

Yksityiskohtaisemmin sanottuna: Laitetaan osoitin taulukon A alkuun ja taulukon B loppuun. Jos osoittimien osoittamien summien summa on alle halutun, siirretään taulukon A osoitinta eteenpäin. Muussa tapauksessa siirretään taulukon B osoitinta taaksepäin ja toistetaan tätä kunnes jompikumpi osoitin menee ulos taulukosta.

Kunniamaininnan ansaitsee **Veikko Sariola**, joka oli kokeillut geneettistä algoritmia MATLABilla, mutta tulos oli vain 3,0997e-05. Osoittautui, että optimoitu raaka voimankäyttö oli paras lähestymistapa ongelmaan.

Voittajaratkaisun listaus verkkojatkoillamme: skrolli.fi/numerot. Skrollin toimituksen blogi: skrolli.fi/blogi.

Tampio – ohjelmoi suomeksi

Miltä näyttää ohjelmointikieli, jonka kielioppi on suomen kirjakieltä? Tampio-kielen kehittäjä Iikka Hauhio kertoo, miten se toimii.

Teksti: Iikka Hauhio Kuvat: Nasu Viljanmaa

Kaikki nykyään yleisesti käytetyt ohjelmointikielet ovat yhdistelmä englannin kieltä ja matematiikasta periytyvää notaatiota. Ohjelmointikielten englanninkieliset sanat eivät kuitenkaan taivu, vaan sanojen väliset suhteet ilmaistaan partikkeleilla. Miksi näin on?

Ehkä syyt ovat historialliset? Jos suomi olisi ollut tiedemaailman valtakieli silloin, kun ensimmäiset ohjelmointikielet kehitettiin, perustuisivatko ne kaikki sijamuotoihin ja verbien modustaivutukseen? Olisivatko avainsanat liitepartikkeleita ja omistusliitteet yleinen tapa viitata eri olioiden kenttiin? Miksi ohjelmointikielet ovat pelkkää erikoismerkkimössöä, jossa harvat englanninkieliset sanat on lyhennetty maksimissaan neljän merkin mittaisiksi?

Tietysti on olemassa syynsä sille, miksi ohjelmointikielet eivät ole täydellistä suomea tai englantia. Luonnollisista kielistä poiketen ohjelmointikielten kielioppi on hyvin tiukka ja määräävä: on yleensä vain yksi tapa sanoa sama asia, synonyymejä ei tunneta, sanajärjestys on pakottava ja yksikin virhe johtaa ymmärrettävyyden menetykseen. Nämä rajoitukset mahdollistavat kuitenkin esimerkiksi sen, että yhdessä lauseessa voi olla mielivaltaisesti sisäkkäisiä lausekkeita.

Ohjelmointikielet ovat vaihtokauppa, jossa väljät ja helpot säännöt on vaihdettu rajattomaan ilmaisuvoimaan taaten, että yksi lause tulkitaan aina samalla tavalla. Hyvin yksinkertaisillakin kielillä voi esittää jokaisen algoritmin, jonka tietokone voi suorittaa.

En kuitenkaan ole näistä perusteluista vakuuttunut. Heti kun huomasin, että on olemassa avoimen lähdekoodin *morfologisia jäsentimiä,* eli luonnollisen kielen sanan taivutusmuodon tunnistavia ohjelmia, päätin alkaa toteuttaa suomenkielistä ohjelmointikieltä, Tampiota. Tahdoin tehdä siitä yhtä aikaa helppolukuisen ja ilmaisuvoimaisen. Tässä artikkelissa kerron kohtaamistani haasteista ja kielestä itsestään.

Moniselitteisyyden ongelmat

Sen lisäksi, että luonnollisessa kielessä sama asia voidaan ilmaista useilla eri tavoilla, myös samat sanat voivat tarkoittaa useita eri asioita. Normaalisti ihmisten välisessä vuorovaikutuksessa moniselitteiset lauseet voi tulkita kontekstin avulla, mutta ohjelmointikielessä tämä on usein vaikeaa.

Eräs käyttämäni esimerkki moniselitteisyydestä on lause ”He tapaavat Jaakon mökillä.” Se voidaan tulkita kahdella tavalla: joko he tapaavat Jaakon jollain mökillä tai he tapaavat Jaakon omistaman mökin luona mahdollisesti *ilman Jaakkoa*. Monitulkintaisuuden lähde on genetiivimuodossa, joka merkitsee suomen kielessä sekä objektia että omistamista. Tietysti asiayhteys avaisi, kuka Jaakko on, mistä mökistä puhutaan ja ketkä tapaavat ketä.

En ole löytänyt Jaakon mökki -ongelmaan tyydyttävää ratkaisua. Tällä hetkellä Tampio-kieleen on toteutettu eräänlainen kompromissi, jossa adjektiiveja käytetään tekemään lauseista yksiselitteisiä.

Kielessäni muuttuja koostuu aina adjektiivista ja substantiivista. Siksi ”he”, ”Jaakko” ja ”mökki” eivät ole sallittuja muuttujanimiä, mutta ”kivat ihmiset”, ”mukava poika” ja ”viihtyisä mökki” ovat. Omistettavat asiat (eli funktiot ja attribuutit) taas eivät saa sisältää adjektiivia. Näiden rajoitusten avulla lause ”He tapaavat Jaakon mökillä” voidaan kirjoittaa uudestaan kahdella tavalla, joista kummallakin on eri merkitys.

”Kivat ihmiset tapaavat mukavan pojan viihtyisällä mökillä.” merkitsee, että he tapaavat Jaakon jollain mökillä.

”Kivat ihmiset tapaavat mukavan pojan mökillä.” merkitsee, että he tapaavat Jaakon omistamalla mökillä.

Se, että jokaisen muuttujanimen on sisällettävä adjektiivi, ei ole kovin kaunis ratkaisu, vaikka adjektiivit hyvin usein itse asiassa selkeyttävätkin koodia. Olen joustanut säännöistä kahdella tavalla. Ensinnäkin yksikirjaimiset muuttujanimet kuten ”x” ja ”y” ovat sallittuja, mikä helpottaa matemaattisten kaavojen kirjoittamista. Toiseksi funktioiden ja attribuuttien nimet voivat sisältää superlatiivisia adjektiiveja,

minkä vuoksi esimerkiksi ”lyhyen listan suurin alkio” on sallittu muoto.

Yksi mahdollinen ratkaisu olisi, että Tampion oikeinkirjoitussääntöjä muutettaisiin tavallisesta suomesta määräämällä, että kaikki muuttujat on kirjoitettava isolla alkukirjaimella. Tällöin siis ”mökki” ei olisi muuttuja ja ”Mökki” olisi. Kielen tarkoitus on kuitenkin olla joka tavalla oikeaa suomea, joten en mielelläni tingi edes oikeinkirjoituksesta.

Sujuvaa ja vähemmän sujuvaa kieltä

Tampio-kielen pitäisi ohjata käyttäjiään lyhyen koodin kirjoittamiseen, koska lyhyet lauseet on helppo jäsentää mielessä ja muistaa. Lisäksi lyhyissä lauseissa esiintyy epätodennäköisemmin moniselitteisyyksiä. Valitettavasti tämä tavoite näyttää kuitenkin mahdottomalta.

Suomen kielen taivutukseen pohjautuva järjestelmä mahdollistaa suuren informaatiotiheyden. Esimerkiksi yksi taivutettu verbi voi ilmaista sekä aikamuodon, persoonan että moduksen muutamalla kirjaimella. Suomen morfeemi–kirjain-suhde on siis suuri. Morfeemeja ovat sanan jakamattomat osat kuten sanavartalo sekä taivutus- ja johdospäätteet.

Jo kymmenen sanan lauseessa voi olla helposti 20 morfeemia, mikä on huomattavasti enemmän kuin mitä esimerkiksi yleensä perinteisen ohjelmointikielen rivillä suositellaan olevan (jos morfeemia vastaa *token* eli ohjelmakoodin jakamaton yksikkö). Esimerkiksi lausekkeessa ”2/3×4” on viisi morfeemia, kun taas sen luonnollisen kielen vastineessa *kaksi jae-ttu-na kolme-lla ja kerro-ttu-na neljä-llä* on kaksitoista (morfeemien rajat merkitty viivoilla).

Yli puolet morfeemeista siis tavallaan valuu hukkaan. Vaikka operaattorien varsinaiset nimet *jae-* ja *kerro-* eivät oikeastaan ole kovin pitkiä, sellaiset turhat morfeemit kuten aikamuotoa ja persoonaa ilmaiseva *-ttu-* kasvattavat merkkimäärää. Ohjelmointikielessä ei tarvita aikamuotoa ja persoonaa. Tampiossa kaikki operaattorit sisältävät tuon saman päätteen *-ttu/-tty*, eikä suomen kielen kolmea muuta mahdollista päätettä käytetä ollenkaan. Pääte voitaisiin siis jättää hyvin jättää pois, mutta silloin koodi ei olisi enää suomea.

Sanojen sijamuodot eivät ole Tampiossa merkityksettömiä, mutta niitä käytetään oikeastaan sen vuoksi, että on pakko, jotta kieli olisi korrektia suomea, eikä sen vuoksi, että niistä olisi jotain erityistä hyötyä. Taulukossa 1 esitellään eri tapoja, joilla sijamuotoja käytetään.

Sijamuotojen ja *-ttu/-tty*-päätteen lisäksi esimerkkilauseeni sisältää vielä yhden ylimääräisen morfeemin: *ja*-sanan. Sitä käytetään laskujärjestyksen muuttamiseksi. Normaalisti Tampiossa operaattori liittyy aina suoraan edelliseen lausekkeeseen, joten *kaksi jaettuna kolmella kerrottuna neljällä* ilman ja-sanaa tulkitaan ”2/(3×4)”, koska *kolme* on ensimmäinen suoraan ennen *kerrottuna*-operaattoria tuleva lauseke. *Ja*-sana muuttaa tätä tulkintaa siten, että *kerrottuna*-operaattori liittyykin edelliseen operaattorilausekkeeseen, joka on *kaksi jaettuna kolmella*. Jos haluttaisiin viitata edellisen operaattorilausekkeen sijasta johonkin vielä aiempaan, on mahdollista käyttää pilkkuja. Eri tapoja muuttaa laskujärjestystä on esitelty listauksessa 1. Koodi on äärimmäisen epäselvää, mutta en toisaalta halua lisätä minkäänlaisia sulkuja kieleen, koska silloin kieli muuttaisi suomen oikeinkirjoitusta.

Suomen kielessä ei ole hyviä tapoja ilmaista laskujärjestystä, sillä luonnollisessa kielessä lausekkeita ei ole tarkoitus laittaa sisäkkäin kovin montaa tasoa. Teksti, jossa näin tehdään, tuntuu kapulakieliseltä. Tämä on yksi syy siihen, miksi lakitekstit ja tekniset tekstit koetaan usein hankalalukuisiksi. Niissä pyritään määrittelemään asiat niin tarkalla tasolla, että luonnollinen kieli ei enää sovellu siihen.

Kieliopin rakennuspalikat

Perinteisesti eräs ohjelmointikielten suunnittelun tavoitteista on ollut kieliopin minimalistisuus, mikä heijastuu muun muassa useimpien kielten funktiokutsusyntaksiin: kaikkia funktioita kutsutaan samalla tavalla, yleensä laittamalla sulut funktion nimen perään ja niiden sisään argumentti ”f(x)”.

Valitettavasti luonnollinen kieli ei toimi niin. Päin vastoin: sama asia voidaan lähtökohtaisesti sanoa useilla eri tavoilla. Selkein jaottelu syntyy niin kutsuttujen *lauseiden* ja *nominilausekkeiden* välillä, jotka molemmat ovat luontevia tapoja ilmaista luonnollisella kielellä funktiokutsujen kaltaisia asioita.

Lauseet ovat palikoita, jotka sisältävät aina verbin. Ne ilmaisevat usein olemista tai tapahtumista, ja ne ovat ehkä luontevin tapa ilmaista tietokoneelle annettavia käskyjä. Esimerkiksi ”Tulosta viesti”, ”Kysy syötettä” ja ”Jaa x syötteen koolla” ovat kaikki lauseita. Verbien lisäksi lause voi sisältää myös muita lauseenjäseniä, jotka ovat useimmiten nominilausekkeita.

Lauseet vertautuvat ohjelmointikielen funktiokutsuihin ja nominilausekkeet taas argumentteihin. Siispä lause

Liimaa pääsi viemäriin.

```
x jaettuna y:llä vähennettynä z:lla lisättynä w:hen     # x/(y-(z+w))
x jaettuna y:llä vähennettynä z:lla ja lisättynä w:hen  # x/((y-z)+w)
x jaettuna y:llä, vähennettynä z:lla ja lisättynä w:hen # ((x/y)-z)+w
x jaettuna y:llä ja vähennettynä z:lla lisättynä w:hen  # (x/y)-(z+w)
```

Listaus 1.

Nominilauseke eli NP

"NP on yhteisnimitys **substantiivilausekkeelle** ja substantiivilausekkeen tavoin toimivalle pronominin tai numeraalin muodostamalle lausekkeelle. NP:itä ovat siis esim. *talo, pienellä tytöllä, Liisan nuorin poika* sekä *minä, tämä, se joka tulee ensimmäiseksi, joku teistä, kolme lasta.*"

"Substantiivilausekkeen **edussanana** on **substantiivi**. Substantiivi muodostaa lausekkeen joko yksin tai erityyppisten laajennusten kanssa —."

http://scripta.kotus.fi/cgi-bin/visktermit/visktermit.cgi

”Tulosta viesti.” voitaisiin ilmaista perinteisellä ohjelmointikielellä muodossa ”print(message)”. Tässä on kuitenkin yksi ongelma: jos funktiokutsu on aina lause ja sen argumentit nominilausekkeita, funktiolle ei voi antaa argumenttina funktiokutsua, kuten vaikka ”print(sqrt(x))”. Näyttää siltä, että lauseilla voi koodata vain funktioita, jotka eivät palauta mitään arvoja.

Ratkaisuni on merkitä Tampiossa yksipaikkaisia matemaattisia funktioita genetiivillä. Yllä kuvattu ”sqrt(x)” on siksi *x:n neliöjuuri*, ja sen tulostamiseksi riittää sanoa ”Tulosta x:n neliöjuuri.”. Genetiivisen funktiokutsun lisäksi toteutin myös essiivimuotoiset (*-na/-nä*) funktiokutsut. Tampion ymmärtämät erilaiset nominilausekkeet on lueteltu taulukossa 1.

Ero lauseiden ja nominilausekkeiden välillä on hierarkinen: lauseet sisältävät lausekkeita eikä toisin päin. Tietoisen valintani vuoksi tämä jaottelu on periytynyt funktiomäärittelyihin seuraavana rajoituksena: lausemuotoiset aliohjelmat, joita sanon metodeiksi, määritellään listana lauseita; lausekemuotoiset aliohjelmat, joita kutsun funktioiksi, määritellään lausekkeiden avulla. Tästä seuraa, että ”metodeita” ei voi kutsua ”funktioiden” sisältä. Olisin ehkä voinut toteuttaa kielen jotenkin muutenkin, mutta tämä ratkaisu lisää mielestäni Tampio-koodin selkeyttä.

Tampiolla kirjoitetut ohjelmat jakautuvat funktionaaliseen ja imperatiiviseen osioon, joista toinen sisältää funktioita ja toinen metodeja. Nimensä mukaisesti funktionaalinen osio on pakko kirjoittaa käyttäen funktionaalista paradigmaa: esimerkiksi silmukoiden sijaan on käytettävä rekursiota, eikä sivuvaikutuksia voi luoda.

```
 1 # Määrittely
 2 Listalla on alkiot.
 3
 4 # Funktiot
 5 Listan pituus on sen alkioiden määrä.
 6 Lista on tyhjä, jos sen pituus on nolla.
 7 Listan pää on sen ensimmäinen alkio.
 8 Loistavan listan häntä on uusi lista, jonka alkiot ovat loistavan listan
alkiot toisesta alkaen.
 9 Lista yhdistettynä kivalla merkillä on
10         riippuen siitä, onko se tyhjä,
11         joko merkkijono ""
12         tai riippuen siitä, onko sen pituus yksi,
13         joko sen pää merkkijonona
14         tai sen pää merkkijonona, yhdistettynä kivaan merkkiin ja
yhdistettynä sen häntään yhdistettynä kivalla merkillä.
15 Listan merkkijonoesitys on se yhdistettynä pilkuilla "," ja
hakasulutettuna.
16 Lista merkkijonona on sen merkkijonoesitys.
17 Merkkijono hakasulutettuna on merkki "[" yhdistettynä siihen ja
yhdistettynä merkkiin "]".
18
19 # Metodit
20 Kun loistavaan listaan lisätään kiva arvo,
21         sen alkioksi lisätään kiva arvo.
22 Kun loistavalta listalta poistetaan pää,
23         sen alkioiksi asetetaan sen alkiot toisesta alkaen.
24
25 # Esimerkkiohjelma
26 Olkoon lyhyt lista uusi lista, jonka alkioita ovat 1, 2 ja 3.
27
28 Kun nykyinen sivu avautuu,
29         se näyttää lyhyen listan merkkijonoesityksen,
30         lyhyeen listaan lisätään 4,
31         lyhyeltä listalta poistetaan pää
32 ja       se näyttää lyhyen listan merkkijonoesityksen.
```

Listaus 2.

Tampio-lauseke	Pseudokoodi	Selitys
X:n nimi X:n neliöjuuri	X.nimi sqrt(X)	Genetiiviä käytetään Tampio-kielessä olion attribuuttien ilmaisemiseen sekä yksiparametristen funktioiden kutsumiseen.
X merkkijonona X pyöristettynä 2 desimaaliin X kerrottuna Y:llä	str(X) round(X, 2) X*Y	Essiivi ilmaisee muutosta. Essiivimuotoisilla funktioilla voi olla kaksi parametria. Myös matemaattiset operaattorit ovat essiivissä.
X:n alkiot X:n ensimmäinen alkio X:n n:s alkio X:n viimeinen alkio	X.alkiot X.alkiot[1] X.alkiot[n] X.alkiot[-1]	Monikkomuotoiset sanat ovat Tampiossa taulukoita. Niiden alkioita voi käsitellä muuttamalla sana yksikköön ja laittamalla sen eteen järjestysluku tai muuttuja.
X:n alkiot toisesta alkaen X:n alkiot neljänteen päättyen	X.alkiot[2:] X.alkiot[:4]	Taulukosta voi valita avainsanoilla alkaen ja päättyen vain osan alkioita.
se, että näytetään teksti ”Hei!”,	() => näytä(”Hei!”)	Se, että -rakenteella merkitään nimettömiä funktioita.
riippuen siitä, onko P, joko X tai Y	P ? X : Y	Riippuen siitä -lauseke palauttaa joko X:n tai Y:n arvon riippuen ehdosta P

Taulukko 1: nominilausekkeet.

Esimerkkiohjelma

Listauksen 2 rivillä 2 määritellään aluksi selvyyden vuoksi lista-luokka, kuten se on määritelty standardikirjastossa. Luokalla on yksi kenttä *alkiot*, joka on monikkomuodon vuoksi taulukko. Tampion sisäänrakennettu taulukkotyyppi on rajoittunut: taulukon sisällä ei esimerkiksi voi olla suoraan taulukoita, sillä on mahdotonta taivuttaa sanaa monikon monikossa. Siksi lista-luokka on hyödyllinen monissa tilanteissa.

Riveillä 5–17 määritellään joukko

Tampio-lauseke	Pseudokoodi	Selitys
tulostetaan kiva viesti nykyiseen konsoliin tulostetaan nykyiseen konsoliin kiva viesti vänkyrä vihollinen menee nykyisen pelaajan lähelle	tulostaa(kiva_viesti, nykyinen_konsoli) vänkyrä_vihollinen.menee(nykyinen_pelaaja)	Tampiossa sijamuotoja ja postpositioita (kuten lähelle) käytetään yksilöimään argumenttien roolit, minkä ansiosta argumenttien järjestyksellä ei ole väliä. Jokaisella argumentilla on oltava eri sijamuoto tai postpositio.
älykäs komponentti aloittaa toimenpiteen, minkä onnistuessa saatu tulos kirjataan lokiin ja minkä epäonnistuessa syntynyt virhe kirjataan lokiin	älykäs_komponentti.aloittaa_toimenpide() .then(saatu_tulos => saatu_tulos.kirjata_lokiin()) .catch(syntynyt_virhe => syntynyt_virhe.kirjata_lokiin())	JavaScriptin asynkroniset kutsut ilmaistaan minkä-sanan avulla.
jos P, niin A, B ja C	If (P) { A; B; C; }	Jos-lauseen sisällä olevat lauseet (tässä A, B ja C) evaluodaan ehdon P pätiessä.
tulostetaan lyhyen listan jokainen alkio	for (x in lyhyt_lista.alkiot) { tulostaa(x) }	Jokainen-avainsanan avulla voi luoda silmukan

Taulukko 2: lauseet.

```
"A murder in the game hall" by "Iikka Hauhio"

The Game Hall is a room. "The hall is calm and dark. It is usually closed
at this time, but the owner agreed to let you be here. There are rows of
computers here."

The computers are scenery in the Game Hall.

The Office is north of the Game Hall. "You are in the owner's office."

The table is fixed in place in the Office. "A wooden table is placed in the
middle of the office."

The documents are on the table. "There are piles of papers on the table."
Understand "papers" as the documents.

Instead of taking the Documents:
        Say "They belong to the owner and you shouldn't take them."
```

Listaus 3: Koodiesimerkki Inform-7 kielestä.

funktioita, joista tärkeimpiä ovat *yhdistettynä* riveillä 9–14 ja *merkkijonoesitys* rivillä 15. Koska funktiot on pakko nimetä joko nominatiivissa (perusmuodossa) tai essiivissä (*-na/-nä*), nimien keksiminen vaatii aina jonkin verran luovuutta. Rivin 16 hassu *hakasulutettuna* on tulos tästä.

Yhdistettynä tekee listasta merkkijonon, jossa jokaisen alkion väliin on laitettu jokin erotin (*kiva merkki*). Tätä operaatiota kutsutaan usein muissa kielissä nimellä join. *Merkkijonoesitys* ja sen alias *merkkijonona* yhdistävät listan pilkuilla ja ympäröivät sen hakasuluilla. Kaikki muut funktiot ovat apufunktioita näille.

Rivillä 6 määritellään ehtofunktio *tyhjä*, joka eroaa muista funktioista siten, että se palauttaa totuusarvon ja sitä voi käyttää *riippuen siitä* -lauseen ehtona, kuten tehdään rivillä 10.

Riveillä 20–23 määritellään metodit, joita käytetään listan muokkaamiseen. Ne kutsuvat sisäänrakennettuja metodeja *lisätään* ja *asetetaan*, jotka muokkaavat *alkiot*-taulukkoa. Näiden metodien avulla listaa voi käyttää jonona, sillä ensimmäinen metodi lisää loppuun uuden alkion ja jälkimmäinen poistaa ensimmäisen.

Lopuksi ohjelmassa on esimerkki lista-luokan käytöstä. Sen pitäisi tulostaa ”[1,2,3][2,3,4]”.

Jatkokehitys

Tampio kaipaa muutoksia, jotta siitä voisi tulla vakavasti otettava ohjelmointikieli. Jos abstraktiotasoa ei halua korottaa, ainoa vaihtoehto lienee viedä syntaksia kohti perinteisiä kieliä. On olemassa joitain luonnollisen ja ohjemointikielen välimuotoja, joista yksi kuuluisimmista on AppleScript, joka on suunniteltu shell-kielten kaltaiseksi skriptikieleksi. Sen kielioppi perustuu prepositioihin siinä missä Tampio käyttää sijamuotoja ja postpositioita. Kielen avulla voi lähettää viestejä eri ohjelmille ja vaikuttaa niiden toimintaan eri tavoin, kuten vaikka käskemällä tiedostojenhallintaohjelmaa irrottamaan levyn: tell ”Finder” to eject disk ”Levyn nimi”. AppleScript ei ole kuitenkaan täysin englantia, vaan esimerkiksi sen lohkot päätetään end-avainsanalla. Tampiossa viimeistä lausetta edeltää aina *ja*-sana. AppleScript myös sallii symbolisten operaattorien kuten + ja – käyttämisen ohjelman englanninkielisten ilmausten seassa.

Paras vaihtoehto Tampion tapauksessa olisikin sallia ehkä jopa mielivaltaisten symbolien määrittely ja sulkujen käyttäminen niiden kanssa. Ratkaisu on itse asiassa tuttu matemaattisesta tekstistä, jossa on tapana sotkea erittäin paljon symbolista kieltä luonnollisen kielen sekaan. Tampiossa ”matemaattiset osiot” olisi tietenkin selkeästi eroteltu suomenkielisestä koodista. Niitä voisi tulkita kuten yksittäisiä sanoja, jolloin niille voisi antaa muun muassa sijapäätteitä, esimerkiksi *X:ää kasvatetaan 2+2:lla*. Tämä vastaa suurin pirtein matemaattisten kaavojen lausumistapaa puhutussa kielessä.

Tärkein haaste on löytää tasapaino symbolisen notaation ja suomen kielen välillä. Uskon, että tämä on mahdollista ja että tarpeeksi hyvällä jatkokehityksellä Tampiosta voi saada niin käyttökelpoisen, että sillä voi kirjoittaa isompiakin ohjelmia.

Luonnollisilla kielillä kerrotaan tarinoita

Vaikka Tampio ei nykymuodossaan olekaan vakavaan käyttöön soveltuva kieli, luonnollisia kieliä ei kannata heti tuomita ohjelmointikelvottomiksi. Historian saatossa on nimittäin

suunniteltu useita ammattilaistenkin käyttämiä luonnollisiin kieliin, lähinnä englantiin, pohjautuvia ohjelmointikieliä. Ehkä hienoin näkemäni on Inform 7 (esimerkki listauksessa 3). Se noudattaa lähes täydellisesti englannin kielioppia, kuulostaa luonnolliselta, ja kaiken lisäksi sen oppiminen on helppoa. Tämä ihme on saavutettu tiukalla rajauksella: kieltä käytetään tekstiseikkailujen luomiseen eikä monimutkaisten algoritmien toteuttamiseen, minkä vuoksi ohjelmat ovatkin pitkälti maisemien ja tapahtumapaikkojen kuvauksia sekä hahmojen välisten vuorovaikutustilanteiden käsikirjoituksia. Toisin sanoen Inform 7:ää käytetään juuri niihin asioihin, mihin luonnollista kieltä yleensäkin käytetään.

Kuten aiemmin jo totesin, ongelma Tampion kaltaisissa ohjelmointikielissä on, että niiden abstraktiotaso on liian matala ainakin verrattuna luonnolliseen kieleen, joka tietyssä mielessä on kaikista abstraktiotasoista korkeimmalla. Luonnollinen kieli on ylimalkaista ja yksityiskohdatonta – se karttaa teknisiä yksityiskohtia ja olettaa kaiken mahdollisen tunnetuksi. Koska kielen käyttäjät eli ihmiset yleensä tietävät jo kaikki yksityiskohdat, mahdollistuu hyvin lyhyiden mutta silti toimivien viestien kirjoittaminen.

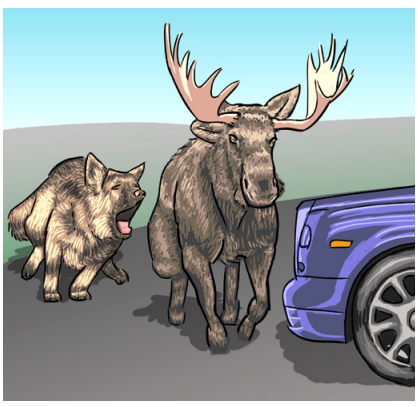
Voin sanoa ohjelmoijalle ”tee minulle blogisivusto” ja olettaa, että hän osaa tehdä minulle jonkinlaisen toimivan sivuston tekstieni julkaisemista varten. Toki todellisuudessa ohjelmoija saa pitkän vaatimuslistan tehtävänannon toteuttamista varten, mutta sekään ei mene samalle tekniselle tasolle kuin ohjelmakoodi.

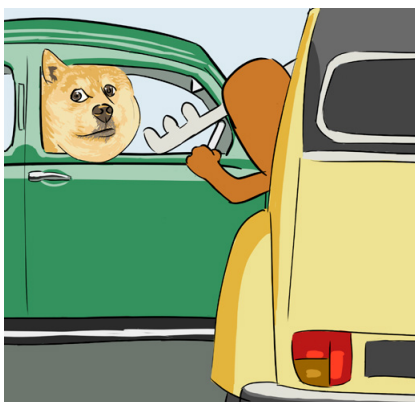
Todellisen suomeen pohjautuvan ohjelmointikielen tulisikin olla sillä tasolla, jolla ohjelmoijia nykyään ohjeistetaan. Sen kääntäjä valitsisi tarpeen mukaan parhaimman mahdollisimman tavan toteuttaa vaaditut ominaisuudet, eikä ”ohjelmoijan” tarvitsisi edes tietää niistä. Kaikki tätä matalamman tason luonnolliseen kieleen perustuvat ohjelmointikielet tulevat aina tuntumaan jossain määrin luonnottomilta, sillä luonnollinen kieli ei yksinkertaisesti sovellu sellaiseen.

Saattaa tuntua scifiltä puhua tällaisista ohjelmoijat korvaavista superkielistä, mutta tämä ei oikeastaan ole kovin kaukainen haave. Ensimmäinen askel olisi luultavasti luoda jonkinlainen käyttötapauskieli, jolla voi kuvata ohjelman ominaisuuksia eräänlaisessa tarinamuodossa. (”Ohjelman aloitusruudulla on lista työntekijöistä. Käyttäjä painaa työntekijän nimeä, ja ohjelma näyttää työntekijän tiedot. Käyttäjä muuttaa työntekijän palkkaa; uusi arvo tallennetaan tietokantaan.”) Kääntäjä loisi käyttötapausten pohjalta ohjelmarungon, jota ohjelmoija voi tämän jälkeen hienosäätää vastaamaan kaikkia vaatimuksia. En usko, että käyttötapauskielen kehittäminen rajatulla sanavarastolla olisi edes kovin vaikeaa. Jos vielä joskus suunnittelen Tampiolle seuraajan, aion ehdottomasti kokeilla luoda sellaisen.

Tampio-kääntäjä Githubissa:
github.com/fergusq/tampio

Toim. huom. Jutussa on käsitelty joitakin kielitieteellisiä käsitteitä hieman yksinkertaistaen ja mutkia oikoen. Tässä muodossaan ne palvelevat Tampio-kieltä, mutta aihepiiristä syvemmin kiinnostuneille lukijoille suositellaan Google-tärpeiksi hakusanoja lausekerakennekielioppi sekä phrase structure grammar.





Koira ajoi hirven auton eteen.

SKROLLI

Skrolli viisi vuotta

Kunnon lehti osaajille

Monet Skrollin pitkäaikaiset lukijat tietävät, miten lehti lähti liikkeelle. Tai ainakin luulevat tietävänsä.

Teksti: Tapio Berschewsky
Kuvat: Tapio Berschewsky, Mikko Heinonen

Vuonna 2012 MikroBitistä uudistui mB. Se mainosti olevansa ”lehti miehille jotka rakastavat vehkeitään” ja julkaisi muun muassa partakonevertailun. Vastareaktio oli vanhojen lukijoiden parissa ymmärrettävän suuri, vaikka samana vuonna mB julkaisi myös yhden DIY-teemaisen numeron.

Matti ”ccr” Hämäläinen yhdessä Skrollin tulevan ensimmäisen päätoimittajan **Ville-Matias ”viznut” Heikkilän** kanssa tekivät uudistuksen innoittamana luonnoksen siitä, miltä vanha kunnon MikroBitti voisi näyttää, jos sitä ei olisi kertaakaan uudistettu.

Viraali-irkkihitiksi noussut kuva keräsi kehuja. Lopulta **Ville ”cos” Rankin** antama palaute sai jotain syttymään Heikkilässä. Tämänhän voisi toteuttaa oikeasti. Nimiehdokkaiden joukossa pyöri *Tritti*. Tarkempi käsittely alkukuukausista löytyy Skrollin numerosta 2015.1.

Idea jostain Skrollin kaltaisesta oli kuitenkin muhinut jo pidempään.

Vielä kertomatonta

Sade pyyhkäisi katua pitkin edestäni eräänä aamupäivänä Helsingissä keväällä 2012. Olin jo tunnin ajan marssinut kuumeisesti tien päästä toiseen. Vaelsin muun muassa Pelaaja-lehteä kustantavan H-Townin toimiston edessä keräämässä rohkeutta.

Suunnitelmanani oli astua kutsumatta sisään ja esitellä ideani uudes-

ta lehdestä firman pomoille. Halusin päästä kirjoittamaan niistä tietotekniikka-aiheista, mistä itse tahdoin lukea juttuja, mutta joita MikroBitti ei enää halunnut julkaista.

Jänistin ja siirryin märkänä kahvilaan jurottamaan. En uskonut, että unelmieni tietokonelehteä koskaan julkaistaisiin.

Lehti, jota en koskaan perustanut, olisi ollut nimeltään *Osaaja*. Se olisi keskittynyt siihen miten tietokoneita voi käyttää, sen sijaan millaisia tietokoneita voi ostaa. Toisin kuin Skrolli, se olisi keskittynyt vain nykytekniikkaan ja siihen mitä sillä voi tehdä, mutta muilta osin kuvittelemani aihekirjo oli hyvinkin pitkälti Skrollin kaltainen.

Myös nykyinen vastaava päätoimittajamme **Mikko Heinonen** oli visioinut jo paljon aiemmin omaa versiotaan tietotekniikkalehdestä, joka tekisi sen, mitä MikroBitti ei. Vuosituhannen alussa Heinonen pyöritteli ajatusta lehdestä, jossa julkaistaisiin asiapitoisella kulmalla kirjoitettuja ohjelmointijuttuja ja teknisestä näkökulmasta tehtyjä peliarvosteluja.

Heinosen läpyskän nimi olisi ollut tamperelaisen ystävälliseen sävyyn *Kunnon Lehti*. Idea lehdestä lämmitti mutta ei kypsynyt lopulta niin pitkälle, että projekti olisi edennyt tuotantovaiheeseen saakka.

Toisin sanoen, jo ennen kuin Skrollista tuli Skrolli, olivat kaikki kolme päätoimittajaamme ehtineet harkitsemaan omilla tahoillaan lehden perustamista.

Kohti nollanumeroa

Kun irkkilähtölaukaus Skrollille oli annettu, liityin heti projektin kanavalle, jossa Heikkilä toivotti minut tervetulleeksi kutsumalla minua teollisuusvakoilijaksi. Todellisuudessa olin välitön käännynnäinen. Liityin toimituksen taustavoimiin ajatuksella, että vaikka en voisi bittiläisenä kirjoittaa lehteen, voisin ainakin opettaa toimitukselle kaiken, mitä olin kahdeksassa vuodessa MikroBitissä oppinut tekniikkalehden tekemisestä.

Aivan alussa toimitus koostui nimittäin Heinosta, taittajaamme **Nasu ”basscadet” Viljanmaata** ja allekirjoittanutta lukuunottamatta pääosin henkilöistä, joilla ei ollut lainkaan kokemusta journalismista. Mukana oli jopa yksi alaikäiseksi laskettava juniori, **Elias ”pumpuli” Linjama**, joka suunnitteli logomme.

Yhteyteni kahden lehden välillä poiki myös tapaamisen Sanomien kanssa, jossa keskusteltiin mahdollisesta yhteistyöstä. Meille vilautettiin muun muassa mahdollisuutta tehdä Skrollia Bitin liitteenä. Se ei onneksi johtanut mihinkään, mutta tilaisuus oli sikäli merkittävä, että Skrollin toimitus tapasi toisensa silloin ensimmäistä kertaa kasvotusten.

Kun päätös itsenäisen paperilehden tekemisestä oli tehty, väänsimme pika-aikataululla parissa viikossa Wärk:Fest-tapahtumaan näytteen, joka oli käytännössä taitettu A3-arkillinen pieniä artikkelinpätkiä, jotka edustivat sitä mitä halusimme tehdä.

Mikko Heinonen tulosti niitä ison kasan yrityksensä ”digipainokoneella”, joka ei ollut kovin halukas painamaan. Valitsemamme paperi oli aivan aavistuksen liian paksua, ja koneessa olikin prosessin aikana parhaimmillaan jopa 7 yhtäaikaista paperitukosta eri puolilla laitetta.

Heinosen uurastukset rakkineen parissa kuitenkin palkittiin, ja saimme Wärk:Festiin ison pinon Skrolli 2012.0 -lehdykkää, jonka sisäiseksi nimeksi muodostui nollanumero. Ajoittain mietin miten monelta nykyiseltä lukijalta tämä aarre löytyy Skrolli-kokoelmasta. Muiden onneksi se ja muut vanhat Skrollit ovat kotisivuillamme sähköisessä muodossa (skrolli.fi/numerot).

Nollanumero sai hyvän palautteen, ja keräsimme kohtuullisen määrän tilaajia jo Wärk:Festeillä. Kun ensimmäinen Skrolli tuskien taipaleen jälkeen oli viimein valmis painoon, olimme jo nelinumeroisessa tilaajamäärässä. Lämmin kiitos kaikille, jotka luottivat.

Skrolli 2013.1 ilmestyi 25.3.2013. Myöhemmin Alma Talent onneksi pelasti ja korjasi sen MikroBitinkin.

Meno ihan Skrolli

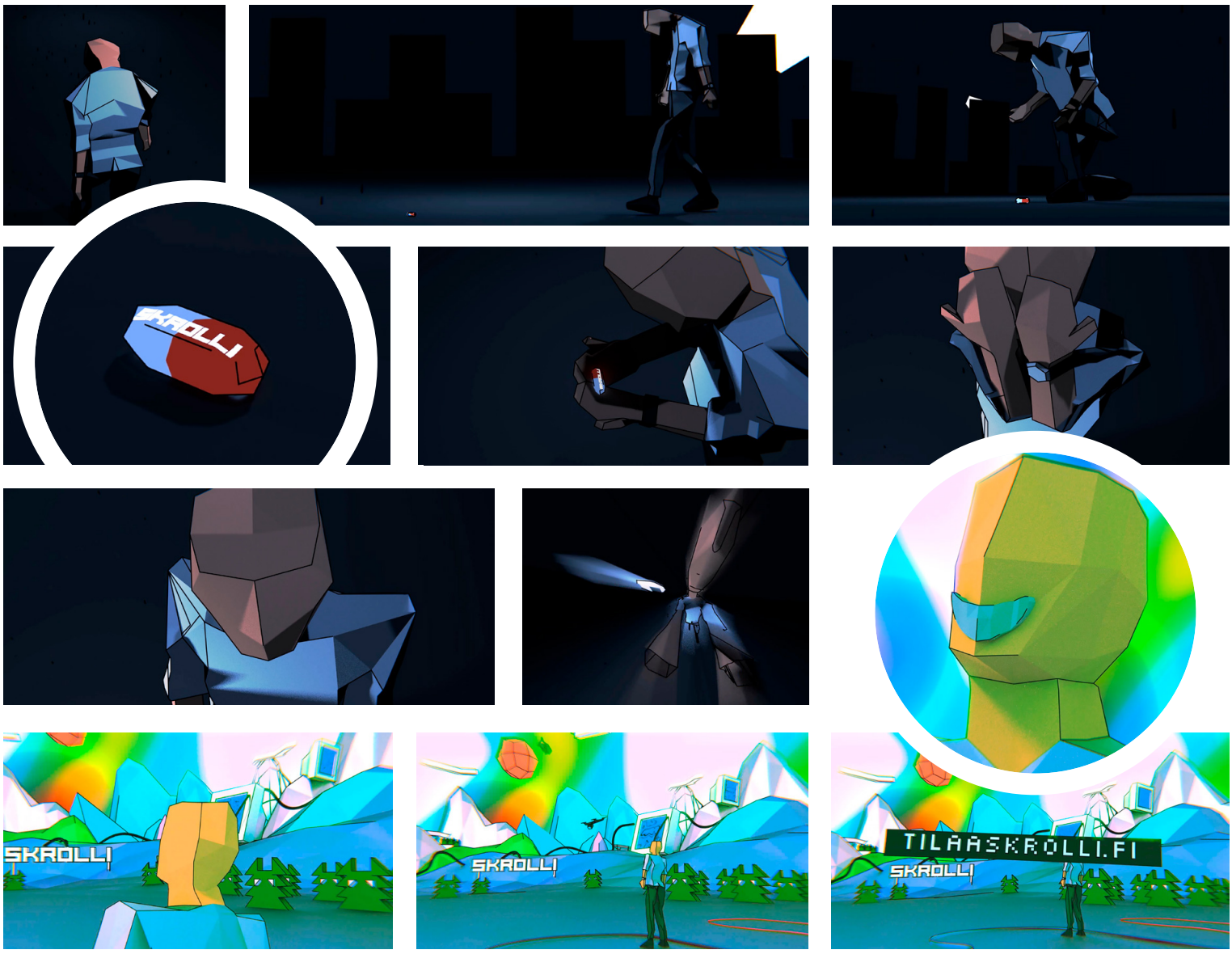
Nykyään Skrollissa puuhastelee useampikin media-alan ammattilainen, muun muassa MikroBitin ikiwanha avustaja ja hiljattain perustetun Retro Rewind -retropelilehden päätoimittaja **Jukka O. Kauppinen** sekä MikroBitti- ja Saku-veteraani **Janne Sirén**.

Tästä huolimatta lehteä tehdään harrastepohjalta, omasta vapaa-ajasta uhraten, silloin kun kukin värikkäästä tiimistämme pystyy, jaksaa ja ehtii. Vaikka lehti saadaan joka kerta ulos ajallaan, varsinkin viimeiset päivät ennen julkaisua ovat usein aika kaoottisia. Mokia tapahtuu, ihmiset ovat väsyneitä, ja kuten toimituksessa sanomme, on ”meno ihan Skrolli”.

Siinä se. Skrollin ensi metrit, vähän eri kulmasta kerrottuna kuin viimeksi. Juhlavuoden aikana palaamme aiheeseen vielä – muuten jatkamme sitä mitä olemme tehneet tähänkin asti.

Lehtiuudistuksia ei siis tarvitse pelätä, vaikka välillä kokeilemme jotain uuttakin: Viime vuonna aloitimme Skrollin blogin, jossa kerromme lehden tekemisestä, niistä mokailuistakin. Tulossa myös tanakkaa yhteisöasiaa – lisää siitä lehden sivulla 2!

Toimituksen blogi: skrolli.fi/blogi



TERVETULOA SKROLLI-KAUPPAAN: TILAASKROLLI.FI

TILAA SKROLLI 2018

alk. **37,99 €**
paperilehti
digilehti
PDF-lehti

IRTONUMEROT

alk. **5 €** paperi / **1,99 €** digi
paperilehtiä 2014-2018
International Edition 2016-2017
digilehdet 2013-2018

SKROLLI-T-PAIDAT

20 €
musta/vihreä
valkoinen/turkoosi
valkoinen/pinkki

SKROLLI

Reaktor 18 Breakpoint

It's happening again

– A one-day conference for digital makers, developers, designers and content creators.

The hugely popular Reaktor Breakpoint conference is happening again for one awesome day on May 24th.
Speed is what you need – a limited batch of discounted early bird tickets are out now, so make sure to grab yours before they sell out.

reaktorbreakpoint.com

6 430057 375747